



UNIVERSITÀ DI PARMA

Dipartimento di Ingegneria e Architettura
Corso di Laurea in Ingegneria Informatica, Elettronica e delle
Telecomunicazioni

Analisi audio per la classificazione di brani musicali tramite clustering e deep learning

Audio analytics for music classification using clustering
and deep learning

Relatore:
Prof. Michele Tomaiuolo

Tesi di Laurea di:
Manuel Tanzi

“Non passo il mio tempo a pontificare su cose di alto concetto; Passo il mio tempo a risolvere problemi di ingegneria e produzione.”

Elon Musk

Indice

Introduzione	1
1 Spotify	3
1.1 Presentazione	3
1.2 Spotify Developer	4
2 Stato dell'arte	5
2.1 Similarità	5
2.1.1 Cosine Similarity	6
2.1.2 Distanza di Manhattan	7
2.1.3 Distanza di Chebyshev	8
2.2 Clustering	10
2.2.1 K-means	11
2.2.2 DBSCAN	12
2.2.3 BIRCH	13
2.3 Analisi dei cluster	14
2.3.1 Varianza	14
2.3.2 Mediana	14
2.3.3 Elbow-Curve	15
2.3.4 Silhouette	15
2.3.5 Calinski-Harabasz	16
2.3.6 Davies-Bouldin	16
2.4 Siamese Neural Network	17

3 Analisi audio	19
3.1 Caratteristiche cepstrali	19
3.1.1 Mel-Frequency Cepstral Coefficients	19
3.2 Hashes di fase	21
3.2.1 Forma spettrale ciclica	21
3.3 Caratteristiche di forma d'onda e tempo	22
3.3.1 Zero Crossing Rate	22
3.3.2 Tempo Histogram	22
3.3.3 Beat Histogram	23
3.4 Esempio visivo	23
4 Creazione del Dataset	24
4.1 MongoDB	24
4.2 Spotify API	25
4.3 Estrazione delle caratteristiche audio	27
4.4 Impronta digitale	28
4.5 NLP sul testo	30
4.6 Problematiche	31
4.6.1 Limiti di velocità di Spotify	31
4.6.2 Spazio in memoria dei dati	31
4.6.3 Diversità della lingua dei testi	32
4.7 Considerazioni	33
4.8 Grafici	34
5 Implementazione	36
5.1 Dataset	36
5.2 Metrica di similarità	40
5.2.1 Metriche di similarità non adatte	40
5.3 Clustering	41
5.3.1 Tempi di elaborazione	42
5.3.2 t-SNE	42
5.3.3 Valutazione	43

5.4	Grafo delle adiacenze	44
5.4.1	Gaussiana e Deviazione standard	45
5.4.2	Random Walking	46
5.5	Siamese Neural Network	47
6	Risultati	49
6.1	Benchmark	52
6.2	Confronto	53
6.2.1	Clustering + Grafo	53
6.2.2	SNN	53
6.3	Grafica	54
6.3.1	Ricerca	54
6.3.2	Raccomandazione	55
	Conclusioni	56
	A Appendice	58
	Bibliografia	59

Introduzione

C’è una quantità sorprendente di contenuti multimediali che vengono scaricati quotidianamente da Internet a causa della crescente popolarità di piattaforme come Netflix, Youtube e Spotify. A partire da marzo 2019, su YouTube vengono guardate più di 500 ore di video al minuto.[1] La disponibilità di milioni di prodotti senza un “sistema efficace” per guidare le persone nel prendere decisioni potrebbe portare confusione, esitazione e peggiorare l’esperienza degli utenti.

Netflix utilizza un sistema di raccomandazione basato sull’apprendimento automatico che tiene conto di fattori come i programmi che gli utenti hanno visualizzato e la loro valutazione, così come il loro comportamento di navigazione. Amazon utilizza un sistema simile, che tiene conto delle attività degli utenti e delle loro preferenze per fornire raccomandazioni personalizzate.

Attualmente, il gigante dello streaming musicale Spotify ha 11 milioni di artisti e creators, 82 milioni di brani e oltre 450 milioni di persone attive mensilmente.[2] Uno dei motivi per cui Spotify è un grande successo tra le altre piattaforme di streaming musicale online è la playlist “Discover Weekly”¹.

Tony Jebara, VP of Engineering e Head of Machine Learning di Spotify ha spiegato il loro framework come un equilibrio tra esplorazione e sfruttamento nel suo “keynote” al TensorFlow World di Santa Clara, in California[3]. Sfruttamento significa fornire raccomandazioni basate sulle precedenti abi-

¹“Discovery Weekly” è pensata per aiutare gli utenti a scoprire nuova musica che potrebbero non conoscere e che potrebbe piacergli. Ogni settimana, la playlist viene aggiornata, quindi è sempre possibile trovare qualcosa di nuovo da ascoltare.

tudini di ascolto. L'esplorazione, d'altra parte, si basa sul coinvolgimento incerto degli utenti e viene utilizzato più come strumento di ricerca per sa-perne di più su come le persone interagiscono con i contenuti suggeriti. Le raccomandazioni di Spotify sono per lo più governate da un sistema di intelligenza artificiale chiamato "Bandits for Recommendations as Treatments" o semplicemente noto come BaRT. [4] Spotify utilizza anche Natural Language Processing² per analizzare notizie, articoli e blog scritti sul web su canzoni o artisti specifici.

L'intelligenza artificiale sta interferendo sempre più con la nostra vita quotidiana mostrando e suggerendo contenuti appropriati alle nostre abitudini e necessità. I sistemi di raccomandazione automatici di oggi agiscono in modo simile ad un amico fidato che ti dà consigli su cosa guardare o ascoltare. Tuttavia, la costruzione di un sistema di raccomandazione efficace senza fare affidamento sui dati degli utenti o sulla popolarità dei brani può essere una sfida, poichè questi dati sono spesso utilizzati per aiutare a comprendere le preferenze degli utenti e a identificare i brani che potrebbero essere più interessanti per loro. Tuttavia, alcune tecniche come il clustering non supervisionato, la creazione di embedding musicali e la creazione di modelli di generazione di musica basati sull'IA stanno emergendo come metodi promettenti per la raccomandazione musicale senza l'uso esplicito dei dati degli utenti.

"In che modo l'intelligenza artificiale può influire sull'esperienza musicale umana, se non fa affidamento sui dati degli utenti o sulla popolarità dei brani? Come possiamo costruire un sistema di raccomandazione musicale che catturi l'essenza dell'arte musicale senza l'aiuto di fattori esterni?"

²NLP (Natural Language Processing, ovvero elaborazione del linguaggio naturale) è una branca dell'intelligenza artificiale che si occupa di far interagire le macchine con il linguaggio umano in modo che possano comprendere, interpretare e generare testo

Capitolo 1

Spotify

1.1 Presentazione

Spotify è una piattaforma di streaming musicale che ha rivoluzionato il modo in cui le persone ascoltano la musica. Fondata nel 2006 in Svezia da Daniel Ek e Martin Lorentzon, Spotify è diventata rapidamente una delle più grandi e popolari piattaforme di streaming musicale al mondo.

La storia di Spotify inizia con la creazione della sua piattaforma di streaming musicale, che consente agli utenti di accedere a una vasta libreria di brani musicali da qualsiasi dispositivo connesso a Internet. Inizialmente, Spotify era disponibile solo in Svezia, ma è presto diventato disponibile in altri paesi in Europa, e poi in Nord America e in altre parti del mondo. Attualmente in 60 paesi.

Spotify si è distinto dalla concorrenza per la sua interfaccia utente intuitiva, la vasta libreria di brani disponibili, la possibilità di creare e dividere playlist personalizzate. Ora conta oltre 345 milioni di utenti attivi mensili e oltre 155 milioni di utenti paganti (dati di Dicembre 2020). [2] Nel corso degli anni, la società ha anche sviluppato funzionalità come la possibilità di ascoltare la musica offline e la funzione "Scopri" che utilizza algoritmi di raccomandazione per consigliare brani e playlist basati sui gusti dell'utente.[5][6]

1.2 Spotify Developer

Le API¹ di Spotify consentono agli sviluppatori di creare applicazioni e servizi che si integrano con la piattaforma Spotify e forniscono accesso ai dati dell’utente e alle funzionalità di Spotify.

Alcune delle principali funzioni delle API di Spotify comprendono:

- Autorizzazione dell’utente: le API di Spotify consentono agli sviluppatori di accedere alle informazioni dell’utente, come i brani ascoltati e le playlist create, solo dopo che l’utente ha autorizzato l’applicazione.
- Ricerca: le API di Spotify consentono agli sviluppatori di effettuare ricerche all’interno della libreria di Spotify, recuperando informazioni sui brani, gli artisti e gli album.
- Playback: le API di Spotify consentono agli sviluppatori di controllare il playback di Spotify, ad esempio per riprodurre, mettere in pausa o saltare un brano.
- Creazione di playlist: le API di Spotify consentono agli sviluppatori di creare e modificare le playlist dell’utente su Spotify.
- Raccomandazioni: le API di Spotify consentono agli sviluppatori di accedere alle raccomandazioni dei brani basate sui gusti dell’utente per offrire un’esperienza personalizzata.

¹API è l’acronimo di ”Application Programming Interface” (Interfaccia di programmazione delle applicazioni), un insieme di regole e specifiche che consentono ad applicazioni, sistemi o servizi di comunicare tra loro. In generale, un’API specifica come un’altra applicazione o sistema può accedere ai dati o alle funzionalità di un’altra applicazione o sistema.

Capitolo 2

Stato dell'arte

2.1 Similarità

La similarità è un concetto fondamentale in campo informatico, che trova applicazione in molteplici ambiti, dall'elaborazione del linguaggio naturale alla ricerca di informazioni, dalla bioinformatica alla visione artificiale. Esistono diversi tipi di approcci per la misura della similarità tra oggetti, dall'uso di algoritmi di confronto basati sull'uguaglianza, all'utilizzo di metodi basati sull'apprendimento automatico, come le reti neurali.

Uno degli utilizzi innovativi della similarità in campo informatico è la sua applicazione alla generazione automatica di contenuti, in cui algoritmi di machine learning sono utilizzati per generare testo, immagini o video in modo autonomo, prendendo come riferimento contenuti esistenti e trovando quelli più simili[7].

Anche la raccomandazione di prodotti o servizi utilizza spesso tecniche di similarità, analizzando il comportamento degli utenti e individuando quelli con preferenze simili per suggerire loro contenuti o acquisti pertinenti. È quindi considerabile uno strumento molto potente per identificare e raggruppare oggetti simili tra loro[8].

2.1.1 Cosine Similarity

La cosine similarity è una misura di similarità tra due vettori, utilizzata spesso nell'ambito dell'elaborazione del linguaggio naturale e dell'apprendimento automatico. Si basa sulla similarità di orientamento tra i vettori, ovvero sulla loro capacità di puntare nella stessa direzione.

La cosine similarity viene spesso utilizzata per confrontare documenti o parole, ad esempio per identificare i documenti che parlano degli stessi argomenti o per trovare sinonimi di una parola specifica. Viene anche utilizzata per raccomandare prodotti o per classificare i dati in base alla loro similarità.

La formula matematica per calcolare la cosine similarity tra due vettori **A** e **B** è la seguente:

$$\cos(\alpha) = \frac{\mathbf{A} \cdot \mathbf{B}}{|\mathbf{A}| |\mathbf{B}|} = \frac{\sum_{i=1}^N A_i B_i}{\sqrt{\sum_{i=1}^N A_i^2} \sqrt{\sum_{i=1}^N B_i^2}} \quad (2.1)$$

dove \cdot rappresenta il prodotto scalare dei vettori e $|\mathbf{A}|$ e $|\mathbf{B}|$ rappresentano le loro norme.

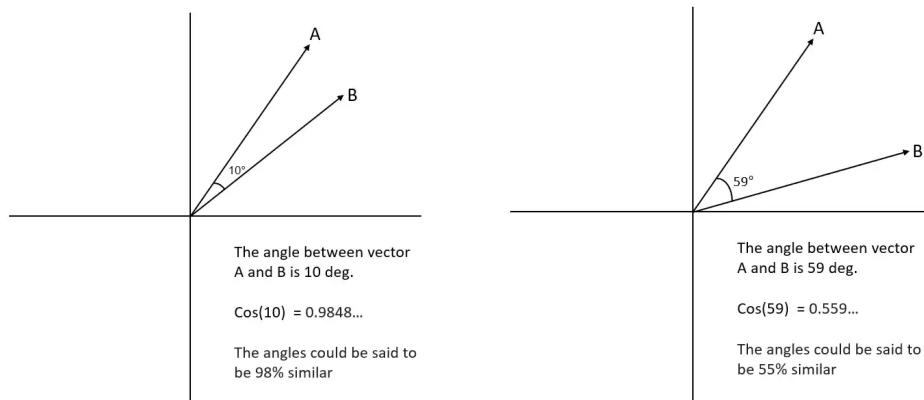


Figura 2.1: Sinistra: Due vettori con una similarità del 98% in base al coseno dell'angolo tra i vettori. Destra: Due vettori con una similarità del 55% in base al coseno dell'angolo tra i vettori.

L'utilizzo della similarità di coseno dispone di diversi vantaggi come: l'influenza della dimensione dei vettori poichè predilige il calcolo dell'angolo tra i vettori stessi invece che la loro distanza, il che significa che può essere utilizzata per vettori con dimensione differente tra loro. Inoltre, non viene influenzata da valori mancanti o zeri e non necessita di alcuna normalizzazione dei dati, quindi è possibile utilizzarlo anche se i dati hanno una scala diversa tra loro. Viene quindi definita una metrica robusta anche in presenza di rumore o dati sparsi. A discapito di quanto elencato sopra cosine similarity assume che le features abbiano la stessa importanza e come detto in precedenza non tiene conto della distanza tra vettori e può dare risultati simili per vettori molto diversi se sono allineati.

2.1.2 Distanza di Manhattan

La distanza di Manhattan, anche conosciuta come distanza L1, è una metrica di distanza che misura la distanza tra due punti in uno spazio multidimensionale come la somma delle differenze assolute tra i valori delle diverse caratteristiche per ogni punto. Tuttavia, a differenza della distanza Euclidea, essa non tiene conto della distanza relativa tra i punti. La formula matematica per calcolare la distanza di Manhattan tra due punti x e y con n caratteristiche è:

$$d(x, y) = \sum_{i=1}^n |x_i - y_i| \quad (2.2)$$

Dove \mathbf{x}_i e \mathbf{y}_i sono i valori delle caratteristiche i -esima per i punti x e y rispettivamente. Il segno di modulo assicura che la distanza sia sempre positiva, questa è una caratteristica della distanza L1.

La distanza di Manhattan è molto robusta in presenza di rumore o valori mancanti nei dati. Tuttavia, può essere influenzata da caratteristiche con intervalli di valori più ampi e può essere più sensibile alle outlier rispetto ad altre metriche.

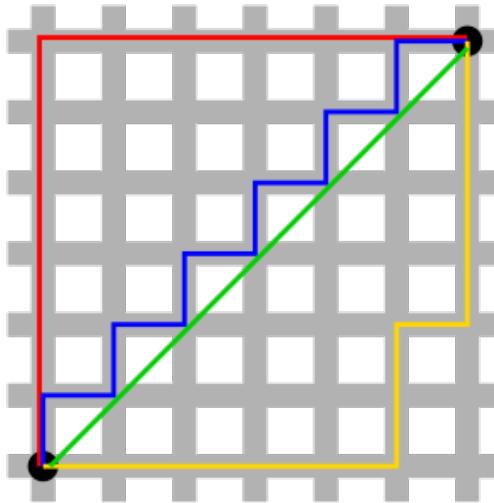


Figura 2.2: [9]Geometria euclidea e geometria del taxi: le linee rossa, blu e gialla nella geometria del taxi hanno tutte la stessa lunghezza (12). La linea verde ha lunghezza $6\sqrt{2} = 8,4853$ nella geometria euclidea, ma continua ad avere lunghezza 12 in quella del taxi (non è quindi più corta delle altre).

2.1.3 Distanza di Chebyshev

La distanza di Chebyshev, nota anche come distanza L_∞ o distanza massima, è una metrica di distanza utilizzata per misurare la dissomiglianza tra due punti in uno spazio multidimensionale. La formula matematica per calcolare la distanza di Chebyshev tra due punti x e y con n caratteristiche è:

$$d(x, y) = \max_{i=1}^n |x_i - y_i| \quad (2.3)$$

dove \mathbf{x}_i e \mathbf{y}_i sono i valori delle caratteristiche i -esima per i punti x e y rispettivamente.

Una delle principali proprietà della distanza di Chebyshev è che essa è molto robusta in presenza di outlier poiché essa si concentra solo sull'elemento più dissimile tra i due punti. In questo modo essa tiene conto della differenza tra ogni singola coordinata tra i due punti, e ne seleziona la maggiore. Questo rende la distanza di Chebyshev una misura efficace rispetto a

piccole differenze tra le coordinate, poichè l'effetto delle differenze più piccole viene “mascherato” dalle differenze più grandi.

D'altra parte, una limitazione della distanza di Chebyshev è che essa può essere eccessivamente influenzata da caratteristiche con intervalli di valori più ampi e può essere meno adatta ai dati normalizzati rispetto ad altre metriche come la distanza Euclidea. L'utilizzo della distanza di Chebyshev dipende dai dati e dall'obiettivo del sistema, ma essa può essere particolarmente adatta per i casi in cui è importante identificare l'elemento più dissimile tra i punti.

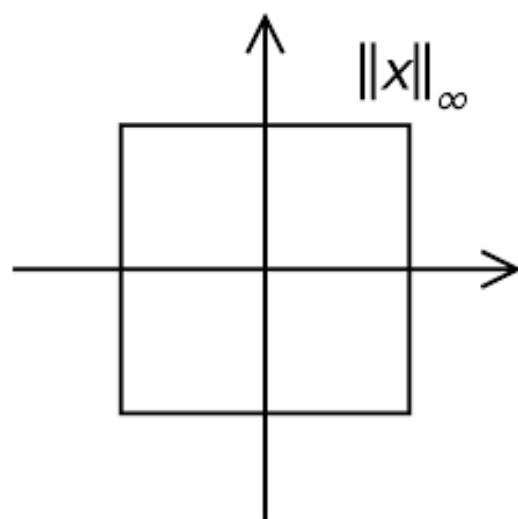


Figura 2.3: [10] Negli scacchi la distanza tra le celle in termini di mosse necessarie al re è data dalla distanza di Čebyšev, da cui il nome.

2.2 Clustering

Il clustering è una tecnica di apprendimento non supervisionato¹ che consente di raggruppare dati in gruppi omogenei in base alle loro caratteristiche. La storia del clustering risale alla statistica e alla teoria dell'inferenza, ma è diventato un campo di studio a sé stante negli anni '50 con l'emergere delle tecnologie informatiche.

Il primo algoritmo di clustering documentato è stato sviluppato nel 1956 da Lloyd. Chiamato k-means, l'algoritmo è ancora ampiamente utilizzato oggi per la sua semplicità e efficienza.[11] Negli anni successivi, sono stati proposti molti altri algoritmi di clustering, tra cui il hierarchical clustering, l'agglomerative clustering, il density-based clustering e il model-based clustering.

Il clustering può essere usato nella scienza dei dati, ad esempio, può essere utilizzato per esplorare i dati e identificare pattern nascosti. Nella bioinformatica, può essere utilizzato per analizzare le relazioni tra le proteine e le loro funzioni. In visione artificiale, può essere utilizzato per raggruppare oggetti simili in un'immagine. In ricerca di mercato, può essere utilizzato per raggruppare i consumatori in base alle loro abitudini di acquisto.

Il clustering è una tecnica importante per l'elaborazione dei dati in quanto permette di sintetizzare informazioni utili dai dati, anche se i dati non sono etichettati. Inoltre, in alcuni casi, può essere utilizzato per ridurre la dimensione dei dati o per creare una rappresentazione compatta dei dati.

¹l'algoritmo non è fornito di etichette e deve scoprire la struttura dei dati in modo autonomo.

2.2.1 K-means

K-means è un algoritmo di clustering che utilizza una tecnica di “partizionamento” per dividere i dati in k cluster. L’algoritmo è basato sull’idea di minimizzare la somma delle distanze quadrate tra i punti di dati e i centroidi² dei cluster.

L’algoritmo inizia inizializzando k centroidi casualmente, quindi assegnando ogni punto di dati al cluster corrispondente al centroide più vicino. Successivamente, i centroidi vengono aggiornati per riflettere la media dei punti assegnati al cluster. Il processo di assegnamento e aggiornamento viene ripetuto fino a quando i centroidi non cambiano più.

La formula utilizzata per calcolare la distanza tra un punto di dati e un centroide è la seguente:

$$d(x, \mu) = \sqrt{\sum_{i=1}^n (x_i - \mu_i)^2} \quad (2.4)$$

dove x è il punto di dati, μ è il centroide del cluster e n è il numero di caratteristiche del punto.

Il criterio di ottimizzazione utilizzato è la seguente:

$$J = \sum_{i=1}^k \sum_{x \in S_i} d(x, \mu_i)^2 \quad (2.5)$$

Dove J è la funzione di costo, S_i è l’insieme di punti assegnati al cluster i-esimo, μ_i è il centroide del cluster i-esimo, $d(x, \mu_i)$ è la distanza tra il punto x e il centroide μ_i .

K-means è un algoritmo semplice e veloce che funziona bene con dati di grandi dimensioni e cluster ben separati, tuttavia, è sensibile alla scelta dei centroidi iniziali e potrebbe non funzionare bene con cluster di forme irregolari o densità variegate.

²Un centroide è un punto rappresentativo di un gruppo di oggetti. Il centroide di un cluster è calcolato come la media aritmetica di tutti i punti di dati assegnati a quel cluster.

2.2.2 DBSCAN

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) è un algoritmo di clustering basato sulla densità. L'idea principale di DBSCAN è quella di raggruppare insieme i punti di dati che sono densamente popolati in uno spazio multidimensionale.

Il principio chiave di DBSCAN è la definizione di una regione densamente popolata come un insieme di punti, dove ci sono almeno un certo numero di punti (MinPts) entro un certo raggio (Eps) dal punto dato. Un punto all'interno di una regione densamente popolata è chiamato “core point”, mentre un punto che non è un core point ma che si trova all'interno del raggio Eps di un core point è chiamato “border point”.

DBSCAN utilizza due parametri: Eps (raggio) e MinPts (numero minimo di punti). Il primo parametro Eps definisce la distanza massima tra due punti per essere considerati come vicini, mentre il secondo parametro MinPts definisce il numero minimo di punti vicini necessari per definire un punto come core point.

DBSCAN utilizza una procedura iterativa per costruire i cluster. Inizia selezionando un punto arbitrario e determinando i suoi punti vicini entro un raggio Eps. Se il numero di punti vicini è maggiore o uguale a MinPts, allora si inizia a costruire un nuovo cluster, aggiungendo tutti i punti vicini e tutti i punti che sono vicini ai punti vicini (e così via). Se un punto non ha abbastanza punti vicini, allora non fa parte di alcun cluster.

DBSCAN è un algoritmo di clustering robusto che è in grado di gestire cluster di forme irregolari e densità variegate, ma è sensibile ai valori dei parametri Eps e MinPts. Inoltre, non è adatto per gestire grandi quantità di dati o dati distribuiti su una griglia.

2.2.3 BIRCH

“**B**” sta per “Balanced” poiché l’algoritmo BIRCH tiene conto della densità dei punti di dati durante la costruzione della struttura gerarchica CF-Tree, in modo da mantenere un equilibrio tra la densità dei punti di dati e la dimensione della struttura.

“**I**” sta per “Iterative” poiché l’algoritmo BIRCH utilizza una procedura iterativa per costruire la CF-Tree, inserendo i punti di dati uno alla volta e suddividendo i nodi interni della struttura quando diventano troppo densi.

“**R**” sta per “Reducing” poiché l’algoritmo BIRCH utilizza la suddivisione dei nodi interni della struttura per ridurre la densità dei punti di dati.

“**C**” in BIRCH sta per “Clustering”, poiché l’algoritmo BIRCH è progettato per identificare e raggruppare i punti di dati in cluster. L’algoritmo utilizza i sottonodi densamente popolati della CF-Tree per costruire i cluster.

“**H**” sta per “Hierarchies” poiché l’algoritmo BIRCH utilizza una struttura gerarchica, la CF-Tree, per rappresentare i dati e identificare i cluster.

L’algoritmo BIRCH inizia creando una radice vuota per la CF-Tree. Quindi, i punti di dati vengono inseriti nella struttura uno alla volta. Se un nodo interno della struttura diventa troppo denso, viene suddiviso in due sottonodi. Il processo di inserimento dei punti di dati e la suddivisione dei nodi interni continuano fino a quando tutti i punti di dati sono stati inseriti nella struttura.

Una volta costruita la CF-Tree, l’algoritmo BIRCH utilizza una procedura di visita per identificare i cluster. La procedura inizia dalla radice della struttura e visita i nodi in modo da identificare i sottonodi densamente popolati. Questi sottonodi vengono quindi utilizzati per costruire i cluster.

BIRCH è un algoritmo efficiente per gestire grandi quantità di dati e si adatta bene ai dati distribuiti in modo non uniforme, tuttavia, è sensibile alla scelta dei parametri di soglia utilizzati per suddividere i nodi della struttura.

2.3 Analisi dei cluster

2.3.1 Varianza

La varianza è una misura della dispersione dei dati intorno alla media.

Viene calcolata come la media del quadrato delle deviazioni dei valori rispetto alla media.

La formula matematica per calcolare la varianza di un insieme di dati è:

$$\sigma^2 = \frac{1}{N} \sum_{i=1}^N (x_i - \mu)^2 \quad (2.6)$$

La varianza può essere utilizzata come criterio di clustering per determinare la compattanza dei cluster.

2.3.2 Mediana

La mediana è il valore che divide un insieme di dati in due parti uguali, con metà dei dati al di sopra e metà al di sotto del valore.

La formula matematica per calcolare la mediana di un insieme di dati è:

$$\text{Se } N \text{ è dispari, mediana} = x\left(\frac{N+1}{2}\right) \quad (2.7)$$

$$\text{Se } N \text{ è pari, mediana} = \frac{x\left(\frac{N}{2}\right) + x\left(\frac{N}{2} + 1\right)}{2} \quad (2.8)$$

La mediana può essere utilizzata come criterio di clustering per determinare la centralità dei cluster.

2.3.3 Elbow-Curve

L’elbow-curve è un metodo per determinare il numero ottimale di cluster in un insieme di dati. Si basa sull’analisi della varianza dei dati all’interno dei cluster.

La formula matematica per calcolare la varianza intracluster è:

$$\text{WCSS} = \sum_{i=1}^N (x_i - \mu)^2 \quad (2.9)$$

L’elbow-curve mostra come la varianza intracluster cambia al variare del numero di cluster. La problematica è che questa tecnica non è sempre facilmente interpretabile, in alcuni casi non c’è un punto netto in cui la varianza intracluster smette di diminuire significativamente.

2.3.4 Silhouette

La silhouette è una metrica utilizzata per valutare la qualità di un clustering.

La formula matematica per calcolare la silhouette di un punto è:

$$\frac{b - a}{\max(a, b)} \quad (2.10)$$

dove a è la distanza media tra il punto e tutti gli altri punti del suo stesso cluster e b è la distanza minima media tra il punto e tutti i punti di un altro cluster.

Valori elevati della silhouette indicano un’alta qualità del clustering.

2.3.5 Calinski-Harabasz

La metrica Calinski-Harabasz è un modo per quantificare la qualità di un clustering. Un'alta valutazione dell'indice Calinski-Harabasz indica una buona separazione tra i cluster e una bassa sovrapposizione. Questo indice può essere utilizzato per scegliere il numero ottimale di cluster per il dataset in questione.

La formula matematica per l'indice di Calinski-Harabasz è data da:

$$\text{Calinski-Harabasz Index} = \frac{\text{Tr}(B)/(k - 1)}{\text{Tr}(W)/(n - k)} \quad (2.11)$$

k è il numero di cluster.

$\text{Tr}(B)$ è la traccia della matrice di covarianza tra i centroidi dei cluster.

$\text{Tr}(W)$ è la traccia della matrice di covarianza all'interno dei cluster.

n è il numero di elementi del dataset.

2.3.6 Davies-Bouldin

L'Indice di Davies-Bouldin (DB Index) è una metrica comunemente utilizzata per valutare la qualità del clustering.

La formula matematica del DB Index è:

$$\text{DB} = \frac{1}{k} \sum_{i=1}^k \max_{1 \leq j \leq k, j \neq i} d(i, j) \quad (2.12)$$

dove k è il numero di cluster, $d(i, j)$ è la distanza tra i centroidi dei cluster i e j , e la funzione max prende la massima distanza tra ogni coppia di cluster. Questa formula viene ripetuta per ogni cluster e il valore finale viene calcolato come la somma di tutti i valori massimi.

L'indice di Davies-Bouldin viene utilizzato per valutare la qualità della separazione tra i cluster, dove un valore basso indica una buona separazione e un valore alto indica una sovrapposizione tra i cluster.

2.4 Siamese Neural Network

Bromley è il primo a proporre l'architettura delle reti neurali siamesi (SNN) come modelli “gemelli” (ovvero identici e con gli stessi pesi) che elaborano due o più input e ne confrontano gli output.[12] L'intuizione centrale alla base di questo concetto è che il confronto dei vettori delle caratteristiche di output evidenzierà le discrepanze tra gli input. Pertanto, questo approccio è vantaggioso nelle applicazioni di firma o riconoscimento facciale.

Un'altra applicazione degli SNN è l'apprendimento della rappresentazione non supervisionato, anche designato come Self-Supervised Learning (SSL). In particolare, è possibile apprendere rappresentazioni da dati non etichettati alimentando variazioni dello stesso input a modelli CNN (Convolutional Neural Network) gemelli e calcolando la somiglianza tra i vettori delle caratteristiche di output. Questa metrica di somiglianza viene utilizzata come funzione di perdita, portando gli SNN ad apprendere rappresentazioni robuste (ovvero, resistere ai disturbi nei dati di input). Questo processo è indicato come apprendimento non supervisionato contrastivo.

Le reti neurali Siamese rappresentano un tipo innovativo di modelli di apprendimento automatico che sono stati sviluppati per la comparazione e la valutazione della somiglianza tra oggetti. Queste reti sono particolarmente efficaci nel riconoscimento di immagini e nel confronto di testo, frasi e vettori. La loro architettura unica rende questi modelli molto adatti a questo tipo di compiti, in quanto permettono di elaborare i dati in modo da produrre una misura quantitativa della somiglianza tra gli oggetti.

Le reti neurali Siamese differiscono da altri modelli di apprendimento automatico in quanto consistono di due reti neurali identiche che vengono allenate simultaneamente per la classificazione di un insieme di dati. Queste due reti ricevono gli stessi input e producono output distinti che vengono utilizzati per calcolare la somiglianza tra gli oggetti. La flessibilità di questo modello lo rende adattabile a una vasta gamma di applicazioni e problemi, rendendolo una soluzione potente per la comparazione di oggetti.[13]

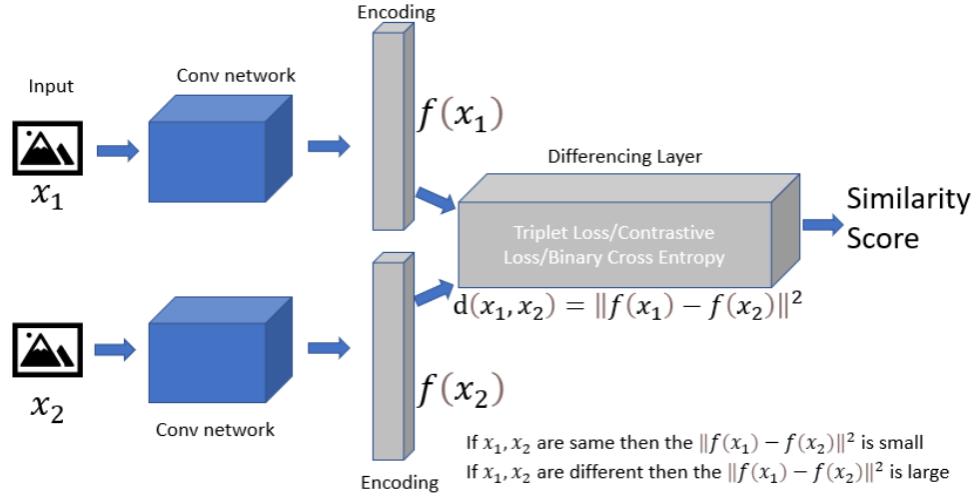


Figura 2.4: [14] Le due sottoreti della rete siamese devono rispecchiarsi a vicenda, quindi devono avere la stessa architettura, parametri e pesi. Queste sottoreti produrranno codifiche per calcolare la differenza tra i due input.

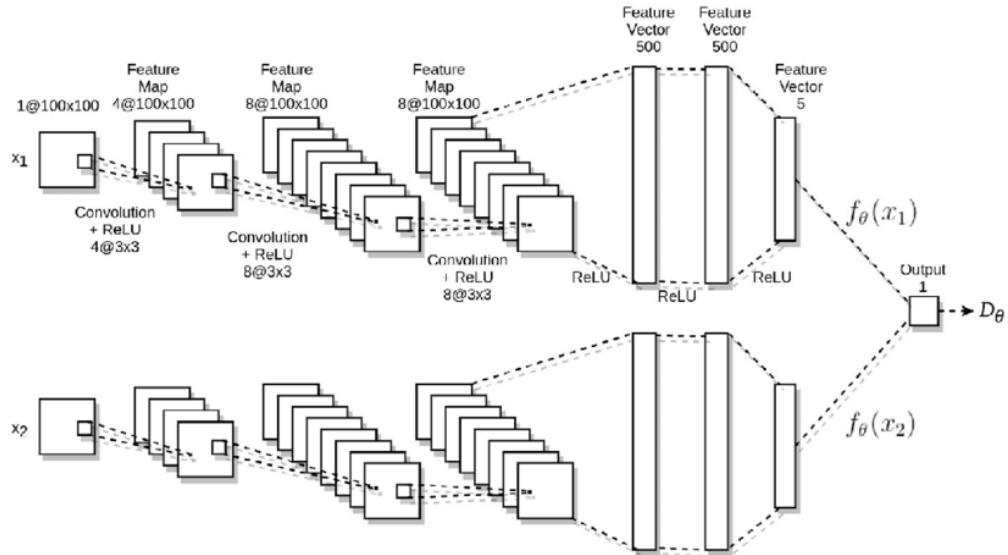


Figura 2.5: Due o più sotto-reti sono create e condividono gli stessi pesi e architettura. L'input viene quindi elaborato dalle sotto-reti in parallelo e l'output viene confrontato per calcolare la similarità tra di essi.

Capitolo 3

Analisi audio

3.1 Caratteristiche cepstrali

Le caratteristiche cepstrali sono una rappresentazione matematica del suono che descrive la distribuzione delle frequenze nel segnale audio. Queste metriche sono ottenute tramite una trasformata matematica nota come trasformata cepstrale, che converte una rappresentazione del suono nello spazio delle frequenze in una rappresentazione del suono nello spazio delle cepstre.

3.1.1 Mel-Frequency Cepstral Coefficients

Il MFCC è una delle caratteristiche cepstrali più comunemente utilizzate per descrivere il suono. Il MFCC descrive il suono come un insieme di coefficienti che descrivono la distribuzione delle frequenze in un file audio.

1. Campionamento del segnale audio
2. Trasformazione del segnale in una rappresentazione delle frequenze con la trasformata di Fourier
3. Mapping delle frequenze sul piano mel, che descrive le frequenze come percepite dall'orecchio umano
4. Calcolo dei coefficienti cepstrali tramite la trasformata cepstrale

- Calcolare la DFT (Discrete Fourier Transform) del segnale audio:

$$X(k) = \sum_{n=0}^{N-1} x_n e^{-j2\pi kn/N} \quad (3.1)$$

dove x_n è il segnale audio campionato, N è il numero di campioni e k è la frequenza.

- Mapping delle frequenze sul piano mel:

$$m(k) = 2595 \log_{10} \left(1 + \frac{k}{700} \right) \quad (3.2)$$

dove k è la frequenza.

- Calcolare la matrice di triangolazione mel:

$$H(i, k) = \begin{cases} 1 & \text{se } f_m(k) \leq f_{mel}(i) < f_m(k+1) \\ 0 & \text{altrimenti} \end{cases} \quad (3.3)$$

dove $f_m(k)$ è la frequenza mel corrispondente alla frequenza k, $f_{mel}(i)$ è la i-esima frequenza mel divisa uniformemente tra 0 e la frequenza di Nyquist e $H(i,k)$ è la matrice di triangolazione mel.

- Calcolare la rappresentazione delle energie delle frequenze:

$$E(i) = \sum_{k=0}^{K-1} |X(k)|^2 H(i, k) \quad (3.4)$$

dove E(i) è la i-esima energia delle frequenze e K è il numero di frequenze.

- Calcolare la trasformata cepstrale:

$$C(i) = \frac{1}{N} \sum_{n=0}^{N-1} \log(E(n)) e^{-j2\pi in/N} \quad (3.5)$$

dove N è il numero di coefficienti cepstrali¹ desiderati e C(i) è il i-esimo coefficiente cepstrale.

¹I coefficienti cepstrali rappresentano una trasformata di Fourier applicata al logaritmo del modulo del coefficiente di Fourier della funzione di autocorrelazione di un segnale.

3.2 Hashes di fase

Gli hash di fase sono spesso utilizzati in applicazioni di riconoscimento audio, in quanto forniscono una rappresentazione stabile e robusta del contenuto audio che non è influenzata da variazioni di volume, pitch o distorsioni.

L'algoritmo di hash di fase consiste nell'effettuare una trasformata di Fourier sul segnale audio, calcolare le proprietà delle fasi del segnale trasformato e quindi utilizzare queste proprietà per creare una rappresentazione univoca del segnale audio. Questa rappresentazione univoca può essere utilizzata per effettuare il confronto tra segnali audio diversi e riconoscere eventuali corrispondenze.

In pratica, le informazioni relative alle proprietà delle fasi del segnale possono essere rappresentate sotto forma di un “hash” a una dimensione, che fornisce una rappresentazione compatta e univoca del segnale audio. Questo hash può essere utilizzato per effettuare il confronto tra segnali audio diversi e determinare se essi rappresentano la stessa canzone o registrazione.

3.2.1 Forma spettrale ciclica

La forma spettrale ciclica (CSS) è una descrizione quantitativa delle informazioni di fase di un segnale audio nel dominio delle frequenze, ed è robusta a variazioni di fase.

La CSS è calcolata come il modulo della trasformata di Fourier (che descrive l'intensità delle componenti di frequenza), trasformato in una rappresentazione circolare:

$$CSS = |\angle X(k)| = \left| \arctan \frac{\text{Im}(X(k))}{\text{Re}(X(k))} \right| \quad (3.6)$$

Dove $\text{Im}(X(k))$ e $\text{Re}(X(k))$ rappresentano la parte immaginaria e reale, rispettivamente, del valore complesso $X(k)$.

3.3 Caratteristiche di forma d'onda e tempo

Le caratteristiche di forma d'onda e di tempo sono una classe di metodi di elaborazione del segnale audio che utilizzano la forma d'onda e la rappresentazione temporale del segnale audio per descrivere le sue proprietà. Queste proprietà possono essere utilizzate per effettuare il confronto tra segnali audio diversi, per la classificazione e il riconoscimento del suono.

3.3.1 Zero Crossing Rate

La Zero Crossing Rate (ZCR) è una metrica che descrive la velocità con cui il segnale audio attraversa lo zero. In altre parole, la ZCR misura il numero di volte in cui il segnale audio passa da positivo a negativo o viceversa in una finestra di tempo determinata.

$$ZCR(n) = \sum_{i=0}^{M-1} [x(n+i) \cdot x(n+i-1) < 0] \quad (3.7)$$

dove $x(n)$ è il segnale audio, M è la dimensione della finestra temporale e $[x(n+i) \cdot x(n+i-1) < 0]$ è un valore booleano che indica se $x(n+i)$ e $x(n+i-1)$ hanno segni diversi.

La ZCR è una metrica molto utile per molte applicazioni audio, in quanto fornisce informazioni sul contenuto del segnale audio, ad esempio se è una voce o un suono musicale, e sulle proprietà temporali del segnale, ad esempio la velocità di ripetizione dei suoni.

3.3.2 Tempo Histogram

Il Tempo Histogram (TH) rappresenta la distribuzione del tempo tra i battiti all'interno di una traccia audio. Può essere calcolato utilizzando la seguente formula:

$$TH(n) = \sum_{i=0}^{M-1} [x(n+i) > T] \quad (3.8)$$

dove $x(n)$ rappresenta l'ampiezza del segnale audio in un dato momento, T è una soglia che viene utilizzata per identificare i battiti e M è il numero di campioni considerati in una finestra di analisi.

3.3.3 Beat Histogram

Il Beat Histogram (BH) è simile al TH, ma tiene conto anche del tempo tra i battiti. Può essere calcolato utilizzando la seguente formula:

$$BH(n) = \sum_{i=0}^{M-1} [x(n+i) > T] \cdot w(i) \quad (3.9)$$

dove $w(i)$ è una funzione di peso che tiene conto del tempo tra i battiti. La funzione di peso può essere definita in base a diverse strategie, ad esempio utilizzando una funzione gaussiana o una funzione a forma di rampa.

3.4 Esempio visivo

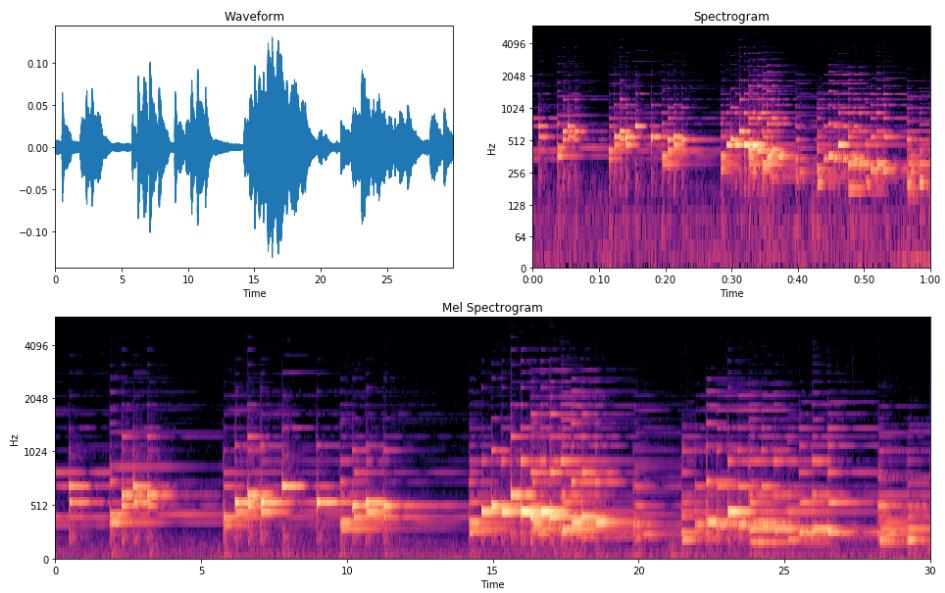


Figura 3.1: [15] Dalla forma d'onda allo spettrogramma per poi ricavare il Mel-spettrogramma

Capitolo 4

Creazione del Dataset

4.1 MongoDB

MongoDB è un sistema di gestione del database NoSQL open source che utilizza un modello di documenti per immagazzinare i dati. MongoDB consente di archiviare grandi quantità di dati non strutturati, come ad esempio immagini, video, messaggi di testo e log, in modo flessibile e scalabile. Invece di utilizzare una struttura di tabelle rigida come in un database relazionale, MongoDB utilizza un formato di documento basato su JSON, che rende più semplice la rappresentazione di dati complessi e la loro gestione.

MongoDB è una delle soluzioni di database più diffuse e utilizzate per sviluppare applicazioni web e mobili, offrendo una vasta gamma di funzionalità avanzate, tra cui la replica, la scalabilità orizzontale e la gestione di cluster.

4.2 Spotify API

L'API¹ di Spotify è un insieme di interfacce software che consentono ai programmati di accedere ai dati e alle funzionalità di Spotify. Attraverso l'API, gli sviluppatori possono accedere a informazioni sulle canzoni, gli album, i generi musicali, le playlist e altri dati correlati a Spotify.

Per utilizzare l'API di Spotify, gli sviluppatori devono registrarsi e ottenere due token di autenticazione che consentono loro di accedere ai dati. Una volta ottenuto i token, gli sviluppatori possono inviare richieste REST² all'API, che restituiscono risposte in formato JSON³.

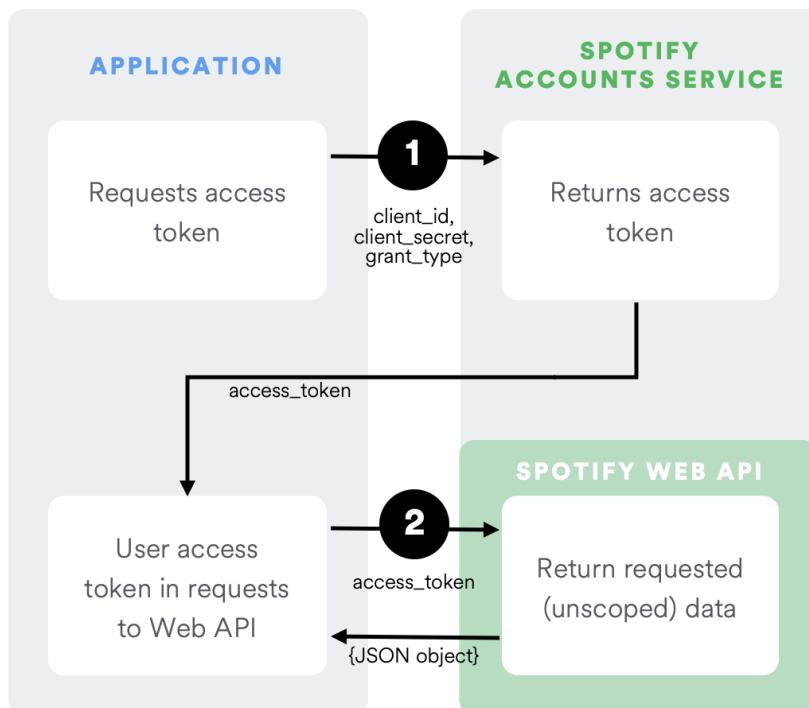


Figura 4.1: Il flusso di credenziali client viene utilizzato nell'autenticazione da server a server. Poiché questo flusso non include l'autorizzazione, è possibile accedere solo agli endpoint che non accedono alle informazioni dell'utente.

¹Application Programming Interface

²Representational State Transfer

³JavaScript Object Notation

Le query utilizzate tra quelle che messe a disposizione da Spotify API sono:

- “`\search`”: Ottiene informazioni sul catalogo di Spotify su album, artisti, playlist, tracce, spettacoli, episodi o audiolibri che corrispondono ad una parola chiave.
- “`\artist\{id\}`”: Ottiene informazioni sul catalogo Spotify per un singolo artista identificato dal suo ID Spotify univoco.
- “`\artist\{id\}\albums`”: Ottiene informazioni sul catalogo Spotify sugli album di un artista.
- “`\albums\{id\}\tracks`”: Ottiene le informazioni del catalogo Spotify sui brani di un album. È possibile utilizzare parametri facoltativi per limitare il numero di tracce restituite.
- “`\tracks\{id\}`”: Ottiene le informazioni del catalogo Spotify per una singola traccia identificata dal suo ID Spotify univoco.
- “`\audio-features\{id\}`”: Ottiene informazioni sulle funzionalità audio per una singola traccia identificata dal suo ID Spotify univoco.
- “`\playlists\{playlist-id\}`”: Ottiene una playlist di proprietà di un utente Spotify.
- “`\playlists\{playlist-id\}\tracks`”: Ottiene tutti i dettagli degli elementi di una playlist di proprietà di un utente Spotify.
- “`\recommendations`”: Le raccomandazioni musicali vengono generate utilizzando le informazioni disponibili riguardo l'artista o la traccia specifica come punto di partenza. Queste informazioni vengono poi confrontate con altri artisti o tracce simili per creare una lista di raccomandazioni. Se ci sono abbastanza informazioni sul punto di partenza specificato, verrà generata una lista di tracce corrispondenti insieme a dettagli sul numero di opzioni disponibili.

4.3 Estrazione delle caratteristiche audio

L'estrazione delle caratteristiche audio è un processo che consente di analizzare un file audio e di estrarre informazioni che possono essere utilizzate per diverse finalità, come la classificazione della musica, la sincronizzazione con altri media e la creazione di effetti sonori. La libreria librosa è una delle librerie Python più popolari per l'estrazione delle caratteristiche audio.

```

y, sr = librosa.load("prova.mp3")

# Calcola i coefficienti Mel-Frequency Cepstral (MFCC)
mfcc = librosa.feature.mfcc(y=y, sr=sr, n_mfcc=13)

# Calcola la matrice di spettro cromatico (CSS)
chromagram = librosa.feature.chroma_stft(y=y, sr=sr)

# Calcola il tasso di incroci zero
zero_crossings = librosa.zero_crossings(y)

# Calcola il tempo e le trame di battito
tempo, beat_frames = librosa.beat.beat_track(y=y, sr=sr)

# Converte le trame di battito in tempi
beat_times = librosa.frames_to_time(beat_frames, sr=sr)

# Calcola la matrice di contrasto spettrale
spectral_contrast = librosa.feature.spectral_contrast(y=y, sr=sr)

# Calcola la matrice di flusso spettrale
spectral_flatness = librosa.feature.spectral_flatness(y=y)

# Calcola la matrice di tempo di decadenza della modulazione
temporal_decay = librosa.feature.delta(mfcc)

# Calcola la matrice di tonnetz
tonnetz = librosa.feature.tonnetz(chromagram=chromagram, sr=sr)

# Calcola la matrice di onde sonore
spectrogram = np.abs(librosa.stft(y))

# Calcola la matrice di onde sonore in dB
db_spectrogram = librosa.amplitude_to_db(spectrogram)

# Calcola la matrice di coefficienti cepstrali
cepstral_coefficients = librosa.feature.mfcc(y=y, sr=sr)

# Calcola la matrice di energia dei coefficienti cepstrali
cepstral_energy = librosa.feature.mfcc(y=y**2, sr=sr)

```

Figura 4.2: Codice Python per estrarre alcune delle features audio utilizzate per l'analisi.

4.4 Impronta digitale

L’impronta digitale audio è una rappresentazione univoca di un file audio che consente di identificare e comparare audio. Le impronte digitali audio vengono create utilizzando caratteristiche estratte dall’analisi del segnale audio, che descrivono la sorgente audio in modo univoco.

Le caratteristiche utilizzate nella creazione di un’impronta digitale audio possono essere diverse a seconda delle esigenze specifiche, ma alcune delle caratteristiche più comunemente utilizzate sono:

- Spectral fingerprint: descrive l’energia spettrale del segnale audio, cioè la quantità di energia in una determinata banda di frequenza.
- Temporal fingerprint: descrive la evoluzione temporale del segnale audio, cioè come la sorgente audio cambia nel tempo.
- Tonal fingerprint: descrive la tonalità della sorgente audio, cioè il dominio della frequenza che governa la sorgente audio.

L’utilizzo di impronte digitali audio nel contesto del machine learning ha molti vantaggi rispetto all’utilizzo di features audio standard. In primo luogo, le impronte digitali audio rappresentano in modo compatto e univoco la sorgente audio, permettendo una facile e rapida comparazione tra file audio. Ciò rende l’utilizzo di impronte digitali audio particolarmente utile nell’identificazione di brani musicali, nella rilevazione di duplicati e nella classificazione di audio.

Inoltre, le impronte digitali audio sono meno suscettibili a disturbi e rumori rispetto a molte delle features audio standard. Ciò significa che le impronte digitali audio sono più affidabili nel riconoscimento e nella comparazione di audio, anche in presenza di rumore o distorsione.

Infine, l’utilizzo di impronte digitali audio permette di risparmiare notevolmente sullo spazio di memoria rispetto all’utilizzo di features audio standard. Poiché le impronte digitali audio rappresentano la sorgente audio in modo molto più compatto, è possibile archiviare un gran numero di file audio

in una quantità molto più piccola di spazio di memoria. Ciò è particolarmente importante quando si tratta di archiviare grandi quantità di dati audio, ad esempio in un database musicale o in un sistema di riconoscimento audio.

In sintesi, l'utilizzo di impronte digitali audio nel contesto del machine learning è molto importante perché offre molti vantaggi rispetto all'utilizzo di features audio standard. Le impronte digitali audio permettono una facile e rapida comparazione tra file audio, sono meno suscettibili a disturbi e rumori e consentono di risparmiare significativamente sullo spazio di memoria.

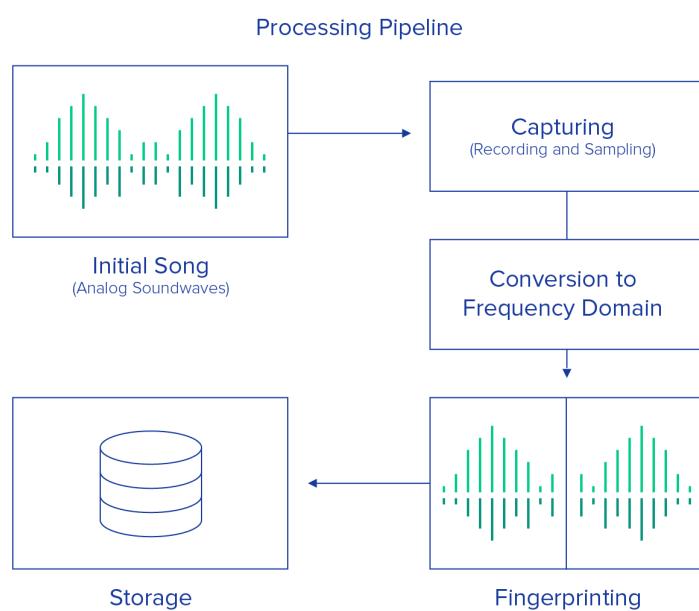


Figura 4.3: Pipeline del processo di salvataggio dati dopo l'analisi audio.

4.5 NLP sul testo

Attraverso la libreria Genius e le API messe a disposizione dalla stessa piattaforma web è possibile ricavare informazioni sulle canzoni, i testi e le traduzioni.

L'analisi del testo di una canzone tramite NLP (Natural Language Processing) può fornire importanti informazioni sul contenuto e lo stile della canzone, nonché sulle tendenze e i modelli della cultura musicale in generale. Ad esempio, l'analisi del testo di una canzone può mostrare il tono e l'emozione dominanti, le parole chiave più frequenti e la struttura linguistica. Queste informazioni possono essere utilizzate per classificare e categorizzare le canzoni, per studiare la creatività e la diversità linguistica nella musica, o addirittura per prevedere il successo commerciale di una canzone.

Se confrontato con l'analisi dei dati musicali come le frequenze sonore, l'analisi del testo di una canzone fornisce informazioni più direttamente legate al significato e alla narrativa, permettendo di capire meglio l'intenzione dell'artista e la risposta del pubblico.

Per effettuare l'analisi NLP dei testi delle canzoni, esistono molte librerie Python disponibili, tra cui NLTK (Natural Language Toolkit), Spacy e TextBlob.



Figura 4.4: Wordcloud che rappresenta le parole che vengono usate con maggior frequenza all'interno dei testi analizzati.

4.6 Problematiche

4.6.1 Limiti di velocità di Spotify

Il limite di velocità dell'API di Spotify è calcolato in base al numero di richieste che vengono eseguite a Spotify in una ripetuta finestra di 30 secondi. Se la tua app supera questo limite di velocità, si iniziano a vedere risposte di errore 429 da parte dell'API Web di Spotify e potresti sentire dagli utenti riguardo un comportamento imprevisto che hanno notato durante l'utilizzo della tua app. Il limite varia a seconda che l'app sia in modalità di sviluppo o in modalità di quota estesa.[16]

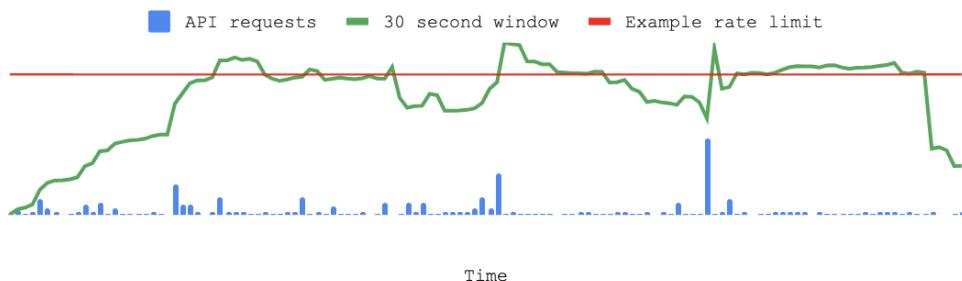


Figura 4.5: L'app riceverà una risposta di errore per i periodi di tempo in cui la finestra di 30 secondi supera il limite di frequenza di esempio in rosso.

4.6.2 Spazio in memoria dei dati

L'analisi audio può diventare una sfida per quanto riguarda la gestione dello spazio in memoria perché i file audio sono spesso molto grandi e richiedono molto spazio di archiviazione. Per questo motivo, quando si effettua un'analisi audio, è necessario ricavare solo le caratteristiche essenziali per poter ridurre la quantità di informazioni da elaborare. Tuttavia, anche la semplificazione dei dati audio attraverso la selezione delle caratteristiche chiave può richiedere una grande quantità di memoria a causa della quantità di informazioni che devono essere gestite. Questo può causare problemi a livello di elaborazione dei dati, specialmente quando si utilizza una tecnologia di ap-

prendimento automatico, che richiede una quantità considerevole di memoria per funzionare correttamente. Pertanto, la gestione ottimale dello spazio in memoria è una considerazione cruciale per l'efficienza e la corretta esecuzione dell'analisi audio.

- **Analisi Audio:** 97.45 GB → 767,953 canzoni
- **Features Spotify:** 8.78 GB → 36,431,278 canzoni
- **Lyrics:** 39.31 GB → 767,953 canzoni

4.6.3 Diversità della lingua dei testi

Nonostante l'enorme potenziale dell'analisi NLP dei testi delle canzoni, ci sono anche alcune problematiche che vanno affrontate. Una delle principali difficoltà è la varietà delle lingue presenti nel dataset. Ad esempio, le canzoni possono essere scritte in lingue diverse come inglese, italiano, spagnolo e cinese, rendendo difficile la traduzione e l'elaborazione delle informazioni. La traduzione può modificare il significato originale del testo e influire sui risultati finali dell'analisi NLP.

Inoltre, il linguaggio utilizzato nella musica spesso include termini e frasi idiomatiche che possono essere difficili da comprendere e analizzare. La natura poetica e immaginativa dei testi delle canzoni può anche rendere difficile l'elaborazione delle informazioni.

Per questi motivi, l'analisi NLP dei testi delle canzoni presenta alcune sfide che devono essere affrontate con attenzione e rigore. E' importante considerare questi aspetti negativi e affrontarli in modo adeguato per garantire che i risultati dell'analisi siano precisi e affidabili.

4.7 Considerazioni

Nel presente paragrafo, si esaminerà un’interessante osservazione riguardante il dataset di canzoni. Dopo aver effettuato un’analisi approfondita, si è scoperto che più della metà delle canzoni presenti nel dataset hanno una popolarità inferiore a 50. Questo dato è significativo perché fornisce informazioni sulle tendenze musicali e sui gusti del pubblico.

Inoltre, questa osservazione suggerisce che c’è una vasta gamma di brani meno popolari che meritano maggiore attenzione e valutazione. Potrebbe essere interessante esplorare queste canzoni meno popolari e capire perché non hanno raggiunto una maggiore popolarità. Questo potrebbe aprire la strada a nuove opportunità per musicisti e produttori musicali, nonché a una maggiore diversità nella scena musicale.

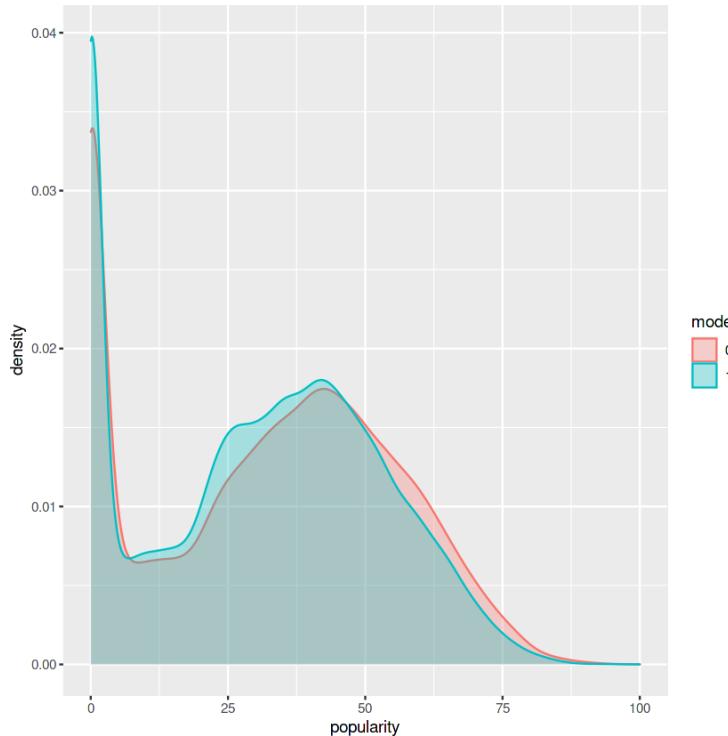


Figura 4.6: “mode” indica la modalità (maggiore o minore) di un brano, il tipo di scala da cui è derivato il suo contenuto melodico. La modalità maggiore è rappresentata da 1 e la modalità minore è 0.

4.8 Grafici

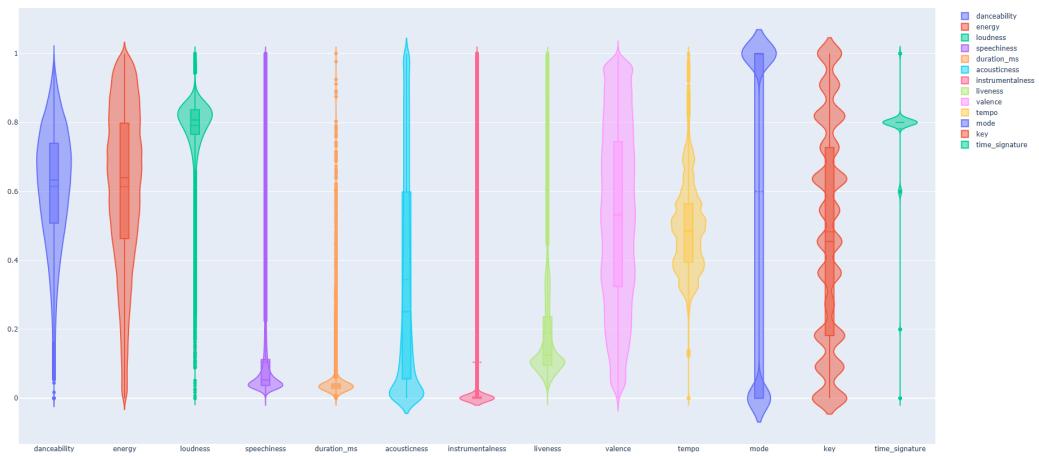


Figura 4.7: Distribuzione dei dati di Spotify normalizzati e scalati tramite Min-MaxScaler con range tra 0 e 1. “Danceability”, “Energy”, “Loudness”, “Speechiness”, “Duration_ms”, “Acousticness”, “Instrumentalness”, “Liveness”, “Valence”, “Tempo”, “Mode”, “Key” “Time_signature”

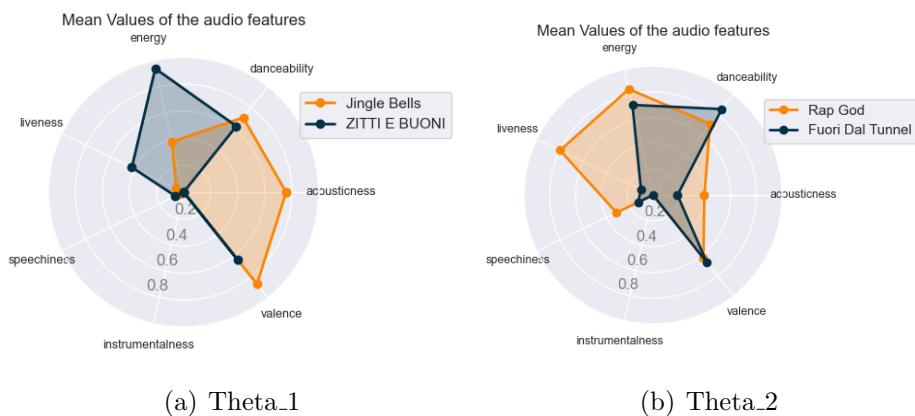


Figura 4.8: In “Theta_1” è rappresentata la griglia theta tra la canzone “Jingle Bells” e “Zitti e buoni”. “Theta_2” invece si ha “Rap God” di Eminem confrontato con “Fuori dal Tunnel” di Caparezza

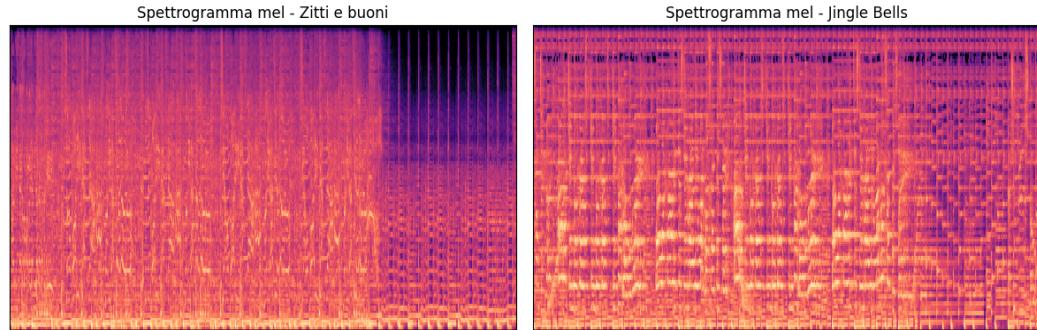


Figura 4.9: Spettrogramma Mel di “Jingle Bells” e “Zitti e buoni”.

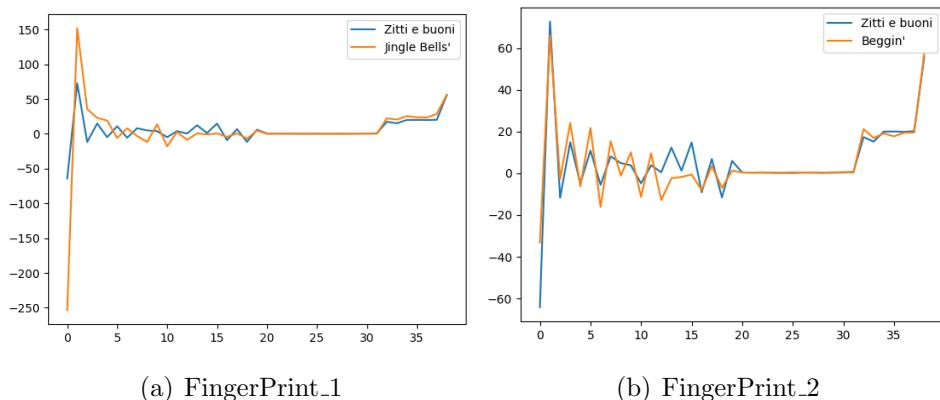


Figura 4.10: Nella figura (a) si mettono a confronto le firme di “Zitti e buoni” e “Jingle bells” e tramite la distanza di Manhattan si ha una similarità di 6.221%. Nella figura (b) si mettono a confronto le firme di “Zitti e buoni” e “Beggin’ ” entrambe del gruppo Maneskin e si nota una similarità dal valore di 92.421%.

Capitolo 5

Implementazione

5.1 Dataset

In primo luogo, si è eseguita un'analisi dei dati, che ha permesso di comprendere la distribuzione dei dati, la presenza di eventuali valori mancanti o duplicati e la qualità dei dati.

Successivamente, è stata eseguita una pulizia dei dati, rimuovendo valori mancanti, duplicati o inaccurati e normalizzando i dati per renderli più adatti per l'utilizzo nei modelli di apprendimento automatico. La normalizzazione dei dati è un passaggio cruciale, poiché molti algoritmi di apprendimento automatico sono influenzati dalle scale dei valori numerici. Pertanto, la normalizzazione aiuta a evitare distorsioni nei modelli che potrebbero derivare da valori anomali o fuori scala.

Sono stati utilizzati differenti scaler, come *StandardScaler*, *MinMaxScaler* e *RobustScaler*, per capire quale fosse il migliore per questo tipo di analisi. Questi scaler sono utilizzati per normalizzare i valori numerici, che possono essere influenzati dalle scale dei valori. *StandardScaler* normalizza i dati sulla base della media e della deviazione standard, portando i valori ad avere media 0 e deviazione standard 1.

MinMaxScaler normalizza i dati sulla base del minimo e del massimo, portando i valori ad avere un intervallo compreso tra 0 e 1.

RobustScaler normalizza i dati sulla base della mediana e dell'intervallo interquartile, che sono statistiche meno influenzate da valori anomali rispetto alla media e alla deviazione standard.

In questo caso specifico, si è scelto di utilizzare *MinMaxScaler* poiché fornisce una normalizzazione che preserva le relazioni tra i valori, rendendola più adatta per modelli che dipendono dalle relazioni tra i valori, come i modelli di clustering o di neuroni artificiali. Inoltre, *MinMaxScaler* è meno influenzato da valori anomali rispetto ad altri scaler come *StandardScaler*.

Il dataset di riferimento che viene utilizzato in questo studio contiene 767,953 di canzoni, ognuna delle quali presenta 16 features differenti:

1. **“*id*”**: L’ID Spotify per la traccia.
2. **“*acousticness*”**: Una misura di fiducia che va da 0.0 a 1.0 sulla base della sicurezza che la traccia sia acustica. 1.0 rappresenta alta fiducia che la traccia sia acustica.
3. **“*danceability*”**: La ballabilità descrive quanto una traccia sia adatta per ballare in base a una combinazione di elementi musicali, tra cui tempo, stabilità del ritmo, forza del battito e regolarità generale. Un valore di 0.0 è meno ballabile e 1.0 è più ballabile.
4. **“*duration_ms*”**: La durata della traccia in millisecondi.
5. **“*energy*”**: L’energia è una misura da 0.0 a 1.0 che rappresenta una misura percettiva di intensità e attività. Tipicamente, le tracce energetiche si sentono veloci, forti e rumorose. Ad esempio, il death metal ha un’alta energia, mentre un preludio di Bach segna basso sulla scala. I tratti percettivi che contribuiscono a questo attributo includono intervallo dinamico, percezione di volume, timbro, tasso di insorgenza e entropia generale.
6. **“*instrumentalness*”**: Prevede se una traccia non contiene voci. I suoni “ooh” e “aah” sono trattati come strumentali in questo contesto.

Le tracce di rap o parola recitata sono chiaramente “vocali”. Più il valore di strumentalità è vicino a 1.0, maggiore è la probabilità che la traccia non contenga contenuti vocali. I valori sopra 0.5 sono destinati a rappresentare tracce strumentali, ma la fiducia è maggiore man mano che il valore si avvicina a 1.0.

7. **“key”:** La chiave in cui si trova la traccia. Gli interi sono mappati sui toni utilizzando la notazione standard di classe di pitch. Ad esempio, $0 = C$, $1 = C\#/\text{Db}$, $2 = D$ e così via. Se non è stata rilevata alcuna chiave, il valore è -1.
8. **“liveness”:** Rileva la presenza di un pubblico nella registrazione. Valori più alti di vivacità rappresentano una maggiore probabilità che la traccia sia stata eseguita dal vivo. Un valore superiore a 0.8 fornisce una forte probabilità che la traccia sia dal vivo.
9. **“loudness”:** Il volume complessivo di una traccia in decibel (dB). I valori del volume vengono calcolati in media sull’intera traccia e sono utili per confrontare il volume relativo delle tracce. La sonorità è la qualità di un suono che è il correlato psicologico primario della forza fisica (ampiezza). I valori in genere variano tra -60 e 0 db.
10. **“mode”:** Indica la modalità (maggiore o minore) di un brano, il tipo di scala da cui deriva il suo contenuto melodico. Maggiore è rappresentato da 1 e minore è 0.
11. **“speechiness”:** Rileva la presenza di parole parlate in una traccia. Più esclusivamente la registrazione è simile al discorso (ad esempio talk show, audiolibro, poesia), più vicino a 1.0 sarà il valore dell’attributo. I valori sopra 0,66 descrivono tracce che probabilmente sono composte interamente da parole parlate. I valori tra 0,33 e 0,66 descrivono tracce che possono contenere sia musica che parole, sia in sezioni che sovrapposte, incluse i casi di musica rap. I valori al di sotto di 0,33 rappresentano probabilmente musica e altre tracce non simili al discorso.

12. ***“tempo”***: Il tempo complessivo stimato di una traccia in battiti al minuto (BPM). Nella terminologia musicale, il tempo è la velocità o il ritmo di un dato brano e deriva direttamente dalla durata media del battito.
 13. ***“time_signature”***: Un tempo in chiave stimato. L’indicazione del tempo (metro) è una convenzione di notazione per specificare quanti battiti ci sono in ogni misura (o misura). Il tempo in chiave va da 3 a 7 indicando tempi in chiave da “3/4” a “7/4”.
 14. ***“valence”***: Una misura da 0.0 a 1.0 che descrive la positività musicale trasmessa da un brano. I brani con valenza alta suonano più positivi (ad es. felice, allegro, euforico), mentre i brani con valenza bassa suonano più negativi (ad es. triste, depresso, arrabbiato).
 15. ***“lng_or”***: Variabile categorica che indica la nazionalità del testo della canzone.
 16. ***“fingerprint”***: Impronta digitale generata in seguito allo studio dell’audio rappresentata da un’array.

```

_id: ObjectId('63ea75f56040eeb4e1a3edd0')
acousticness: 0.649
danceability: 0.656
duration_ms: 203013
energy: 0.648
id: "1IhyIwURwE47j5ZABC24tx"
instrumentalness: 0
key: 11
liveness: 0.164
loudness: -4.789
mode: 1
speechiness: 0.132
tempo: 171.914
time_signature: 4
valence: 0.777
lng_or: "en"
▶ fingerprint: Array
***** Reading from the file ..Database\Database.csv *****
it has 767953 rows and 16 columns
***** It has the following columns *****
Index(['acousticness', 'danceability', 'duration_ms', 'energy', 'id',
       'instrumentalness', 'key', 'liveness', 'loudness', 'mode', 'name',
       'speechiness', 'time_signature', 'valence', 'lng_or', 'fingerprint'],
      dtype='object')

***** Description of quantitative columns*****
             acousticness  danceability  duration_ms   energy    id
count    767953.000000 767953.000000 ... 767953.000000 767953.000000
mean     0.343066    0.613121 ... 0.390857  0.525724
std      0.134042    0.049442 ... 0.045145  0.045145
min      0.000000    0.000000 ... 0.000000  0.000000
25%     0.056100    0.507000 ... 4.000000  0.324000
50%     0.190000    0.650000 ... 5.000000  0.450000
75%     0.584000    0.738000 ... 4.000000  0.744000
max      0.986000    0.989000 ... 5.000000  1.000000
[8 rows x 13 columns]

***** Description of categorical columns*****
           id  lng_or
count    767953 767953
unique   1IhyIwURwE47j5ZABC24tx  en
top      1IhyIwURwE47j5ZABC24tx  en

```

(a) MongoDB

(b) DataProfile

Figura 5.1: Nella figura (a) un esempio di come viene effettuato il salvataggio di una canzone all'interno di Mongodb.

Nella figura (b) il profilo dettagliato del dataset.

5.2 Metrica di similarità

La scelta della tecnica di similarità da utilizzare per la valutazione della somiglianza tra brani musicali è un aspetto cruciale nella creazione di algoritmi di raccomandazione musicale e nell'analisi dei dati musicali. Tra le diverse tecniche di similarità utilizzate per la valutazione della somiglianza tra brani musicali, la *distanza di Manhattan* si è dimostrata particolarmente efficace grazie alla sua capacità di calcolare la distanza lungo gli assi cartesiani e di funzionare bene con dati sparsi e non normalizzati.

Il dataset FMA [17] [18] è stato utilizzato come set di dati di riferimento per il confronto delle performance delle diverse tecniche di similarità. Il confronto ha permesso di valutare la precisione e l'affidabilità delle diverse tecniche di similarità e di determinare che la distanza di Manhattan è stata la migliore in termini di performance e di capacità di fornire risultati accurati e coerenti con le caratteristiche dei brani musicali.

Inoltre, la distanza di Manhattan si è dimostrata particolarmente utile nella valutazione della somiglianza tra brani musicali che presentano differenze di intensità tra le diverse feature. Questo perché, a differenza di altre tecniche di similarità come la cosine similarity, la distanza di Manhattan non è influenzata dalle differenze di intensità tra le feature e consente di valutare la somiglianza tra brani musicali basandosi solo sulla distanza tra i loro punti nel piano cartesiano delle feature.

5.2.1 Metriche di similarità non adatte

La cosine similarity potrebbe essere influenzata dalle differenze nella lunghezza dei brani, il che significa che due brani musicali con la stessa struttura armonica ma con lunghezze diverse potrebbero essere considerati meno simili rispetto a due brani con strutture armoniche diverse ma con lunghezze simili. D'altra parte, la Chebyshev potrebbe sovrastimare l'importanza di alcune caratteristiche, penalizzando le altre, e potrebbe non essere in grado di distinguere tra differenze significative e insignificanti.

5.3 Clustering

Nel corso della ricerca e dell’analisi dei dati del dataset, non è stato possibile identificare alcun valore numerico o categorico che potesse essere utilizzato come etichetta.

Nel contesto di Spotify, infatti, l’etichetta genere viene assegnata agli artisti e non alle singole canzoni. Tale etichetta corrisponde ad un array categorico.

In questo scenario, si fa ricorso a differenti metodi di clustering, tra cui: ***K-means***, ***BIRCH*** e ***DBSCAN***, al fine di condurre un’analisi non supervisionata delle canzoni e quindi studiarne la suddivisione, al fine di confrontare i risultati ottenuti con le etichette di genere.

In particolare, durante lo studio dei dati, sono state utilizzate diverse metriche, come la *mediana*, la *varianza*, l’*elbow-curve*, la *silhouette*, e la *Calinski-Harabasz* e *Davies-Bouldin*, che hanno permesso di determinare il numero di cluster più appropriato da utilizzare.

La ***mediana*** è stata utile per capire la distribuzione dei dati all’interno di ciascun cluster, mentre la ***varianza*** ha permesso di valutare la dispersione dei dati. L’***elbow-curve***, invece, ha consentito di individuare il punto di flessione nella curva della somma dei quadrati degli errori (SSE), indicando il numero di cluster in grado di fornire una buona suddivisione dei dati.

La ***silhouette*** è stata utilizzata per valutare la coesione e la separazione tra i cluster, mentre ***Calinski-Harabasz*** e ***Davies-Bouldin*** ha permesso di valutare la dispersione dei cluster e la loro omogeneità.

Grazie all’utilizzo combinato di queste metriche, è stato possibile individuare il numero di cluster più adeguato per l’analisi in questione. In seguito, una verifica a posteriori dei risultati ottenuti ha confermato che il numero di cluster individuato inizialmente era corretto.

In conclusione, l’utilizzo di queste metriche si è rivelato di fondamentale importanza per l’analisi dei dati e la corretta suddivisione dei cluster, fornendo informazioni cruciali per la comprensione dei dati e l’ottenimento di risultati accurati.

5.3.1 Tempi di elaborazione

La seguente tabella riporta i tempi di esecuzione dei vari algoritmi di clustering impiegati per identificare i cluster di un dataset contenente 767,953 canzoni. La tabella è stata creata utilizzando un computer dotato di processore Intel i5-6400, scheda grafica Nvidia 1050Ti e 16 GB di memoria RAM.

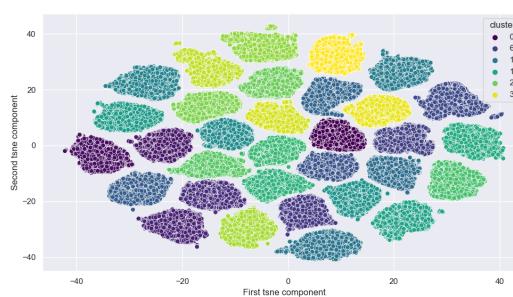
Tra i diversi algoritmi utilizzati, l'algoritmo di K-means ha mostrato il miglior rendimento in termini di tempo di esecuzione, fornendo quindi i risultati in maniera più rapida rispetto agli altri algoritmi considerati.

Tabella 5.1: Tabella dei tempi di elaborazione e predizione

	Fit	Predict
K-Means	121.02 sec	6.43 sec
Birch	197.27 sec	9.89 sec
DBSCAN	57600 sec	57600 sec

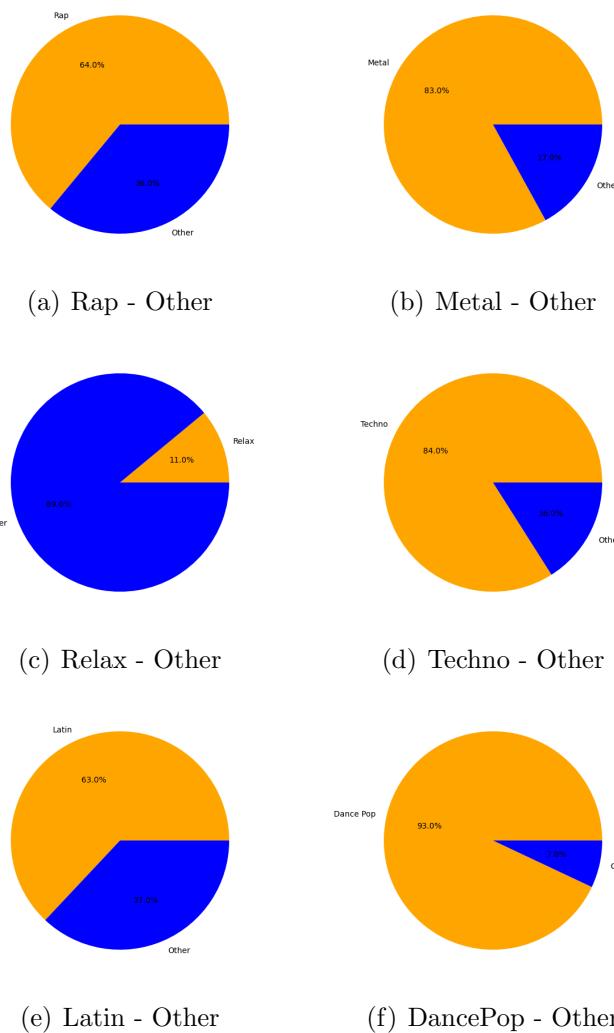
5.3.2 t-SNE

Utilizzando t-SNE (t-distributed Stochastic Neighbor Embedding) per visualizzare i cluster ottenuti, è possibile avere una migliore comprensione della struttura dei dati, identificando eventuali sovrapposizioni o pattern che potrebbero essere difficili da individuare in modo astratto. Ciò potrebbe essere utile, ad esempio, per identificare sottogruppi all'interno dei cluster, o per valutare la validità dei risultati di clustering.



5.3.3 Valutazione

Dopo aver eseguito vari test e confronti, si è riscontrato che l'algoritmo di clustering più idoneo per questo studio è **BIRCH**. Nonostante richieda un tempo di esecuzione accettabile, consente di ottenere la migliore suddivisione delle canzoni in base ai generi classificati da Spotify, fornendo così un valore numerico utile come etichette per le elaborazioni e gli studi successivi. I grafici mostrano alcuni esempi di classificazione per genere all'interno dei vari cluster.



5.4 Grafo delle adiacenze

L'utilizzo della metrica di distanza per costruire un grafo delle adiacenze è un processo fondamentale nell'analisi dei dati. Questo processo coinvolge la rappresentazione di un insieme di oggetti, in questo caso canzoni, come nodi in un grafo, dove gli archi rappresentano le relazioni tra le canzoni. In particolare, l'obiettivo è quello di trovare le canzoni che sono più simili tra loro rispetto ad altre canzoni del dataset.

In primo luogo, è necessario definire il cluster di cui si vuole costruire il grafo. Ciò significa che si selezionano solo le canzoni che appartengono a un determinato cluster dal dataset. Una volta selezionato il cluster, si calcola la metrica di distanza tra le canzoni di questo cluster utilizzando la distanza di Manhattan. Questo permette di ottenere una matrice delle distanze, che rappresenta la distanza tra ogni coppia di canzoni all'interno del cluster.

Per la creazione della matrice delle adiacenze è necessaria solamente una metà della matrice e non tutta poiché è una rappresentazione simmetrica del grafo, ovvero esistendo un arco che va dal nodo A al nodo B, esiste anche un arco dal nodo B al nodo A. Di conseguenza, la matrice delle adiacenze è simmetrica rispetto alla diagonale principale.

In pratica, per ogni *coppia di nodi* (i,j) , la distanza tra i e j è la stessa della distanza tra j e i. Quindi, se si considerasse tutta la matrice delle distanze, si otterebbe una matrice simmetrica rispetto alla diagonale principale, con molte ripetizioni inutili. Per questo motivo, per ottenere la matrice delle adiacenze, è sufficiente prendere solo una metà della matrice delle distanze, ad esempio quella sotto la diagonale principale.

Questo permette di risparmiare tempo computazionale e di memoria, poiché è possibile evitare di calcolare le distanze due volte e di non dover memorizzare i valori duplicati nella matrice delle adiacenze. Inoltre, la direzione degli archi in un grafo delle adiacenze è bidirezionale, quindi è sufficiente considerare solo una metà della matrice per rappresentare correttamente tutte le relazioni tra i nodi del grafo.

5.4.1 Gaussiana e Deviazione standard

Dopo aver calcolato la matrice delle distanze di Manhattan, si procede alla generazione del grafo delle adiacenze. Questo grafo viene costruito rappresentando ogni canzone come un nodo e creando un arco tra due nodi se la distanza tra le rispettive canzoni è inferiore a un determinato valore di soglia. Questo valore di soglia viene calcolato utilizzando la media gaussiana e la deviazione standard calcolate sulla matrice delle distanze.

La **media gaussiana** è una misura di centralità che fornisce il valore medio della distribuzione dei dati, in questo caso delle distanze. La media viene calcolata pesando ogni valore delle distanze in base alla sua distanza dalla media, utilizzando una funzione gaussiana. Ciò consente di ottenere un valore di soglia che tiene conto sia delle distanze più grandi che di quelle più piccole, migliorando la robustezza del metodo.

La **deviazione standard** è una misura di variabilità che indica quanto i dati sono distribuiti intorno alla media. Viene utilizzata insieme alla media gaussiana per determinare il valore di soglia in modo che sia significativo rispetto alla distribuzione dei dati.

$$\text{Valore Soglia} = \text{Media} - \text{Deviazione standard} \quad (5.1)$$

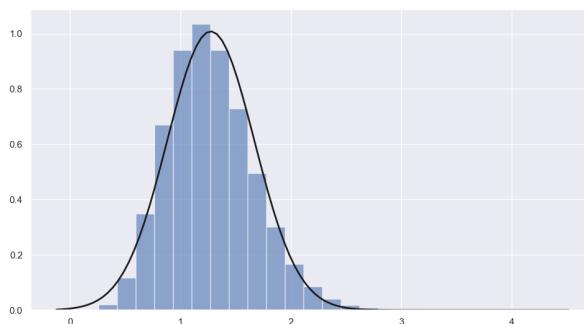


Figura 5.2: Nel grafico viene preso in esempio il cluster con la concentrazione maggiore di brani Rap. Il valore medio risulta 1.27553 mentre la std 0.39568

5.4.2 Random Walking

L’algoritmo di *random walking* viene utilizzato per trovare le canzoni più simili ad una data canzone all’interno del grafo delle adiacenze appena costruito e si basa sull’idea di camminare casualmente sul grafo, partendo dalla canzone di input e muovendosi verso i nodi adiacenti, in modo da identificare le canzoni più vicine ad essa.

L’utilizzo del random walking presenta numerosi vantaggi. In primo luogo, permette di esplorare l’intero grafo delle adiacenze, in modo da identificare le canzoni più simili all’input anche se non sono direttamente connesse ad essa. Inoltre, il random walking è un **algoritmo euristico**, ovvero che non garantisce la soluzione ottimale ma è in grado di fornire soluzioni “sufficientemente” buone in tempi ragionevoli.

In particolare, per utilizzare il random walking, si seleziona la canzone più simile alla canzone di input sulla base della distanza di Manhattan, e si esegue il random walking sui nodi adiacenti a questa canzone, fino a quando non si raggiunge un nodo di destinazione. Questo processo viene ripetuto per un numero fisso di iterazioni, in modo da identificare le canzoni più simili all’input all’interno del grafo delle adiacenze.

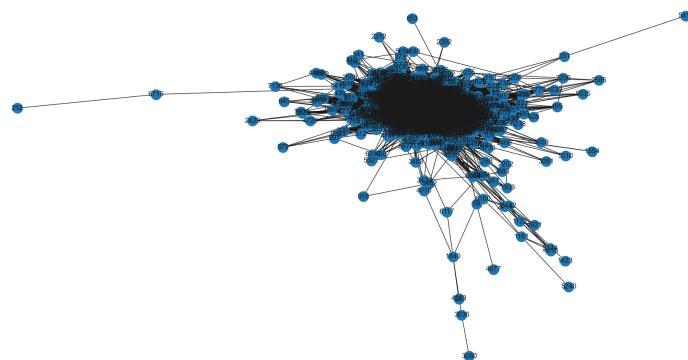


Figura 5.3: Nel grafico viene preso in esempio il cluster con la concentrazione maggiore di brani Rap. Viene raffigurato un tipico grafo delle adiacenze in cui sono presenti solamente gli archi che presentano una distanza inferiore o uguale al valore di soglia.

5.5 Siamese Neural Network

Durante questo studio si è utilizzata un'architettura di rete neurale chiamata Siamese Neural Network per la valutazione della similarità tra brani musicali. L'obiettivo era quello di migliorare la capacità di identificare relazioni non lineari tra brani simili, a differenza di un semplice algoritmo di clustering seguito da un grafo delle adiacenze. L'utilizzo di una Softmax come layer di output ha permesso di avere una percentuale di similarità rispetto ad altri cluster, il che ha esteso le opzioni per la predizione di brani simili.

Le features estratte vengono passate attraverso una serie di layer di **convoluzione** e **pooling**. Questi layer sono responsabili di estrarre le caratteristiche più significative dai dati di input. In seguito, le features estratte vengono concatenate e passate attraverso un layer **completamente connesso** per la creazione di una rappresentazione compatta dell'input.

L'importanza dell'input a coppie di canzoni nella rete neurale siamese consiste nella possibilità di valutare la similarità tra due canzoni invece di cercare di classificare una canzone in un unico cluster. In altre parole, sebbene l'obiettivo sia comunque quello di raggruppare le canzoni in base alla loro similarità, la scelta di utilizzare una rete neurale siamese consente di valutare la similarità tra le canzoni in modo più preciso rispetto all'utilizzo di un algoritmo di clustering seguito da un grafo delle adiacenze.

Per quanto riguarda la metrica di distanza utilizzata, quella di Manhattan è stata scelta per la sua capacità di valutare la differenza tra le caratteristiche audio di due canzoni. Infine, il layer di output è costituito da una **softmax**, che fornisce una probabilità di appartenenza a un cluster per ciascuna delle due canzoni in input.

In generale, l'implementazione di una rete neurale siamese per l'analisi della similarità tra canzoni presenta numerosi vantaggi rispetto ad altre tecniche. Ad esempio, l'uso di una softmax come layer di output consente di superare alcune delle limitazioni degli algoritmi di clustering tradizionali, come la necessità di circoscrivere le canzoni all'interno di un unico cluster, ampliando così la possibilità di scelta per l'utente.

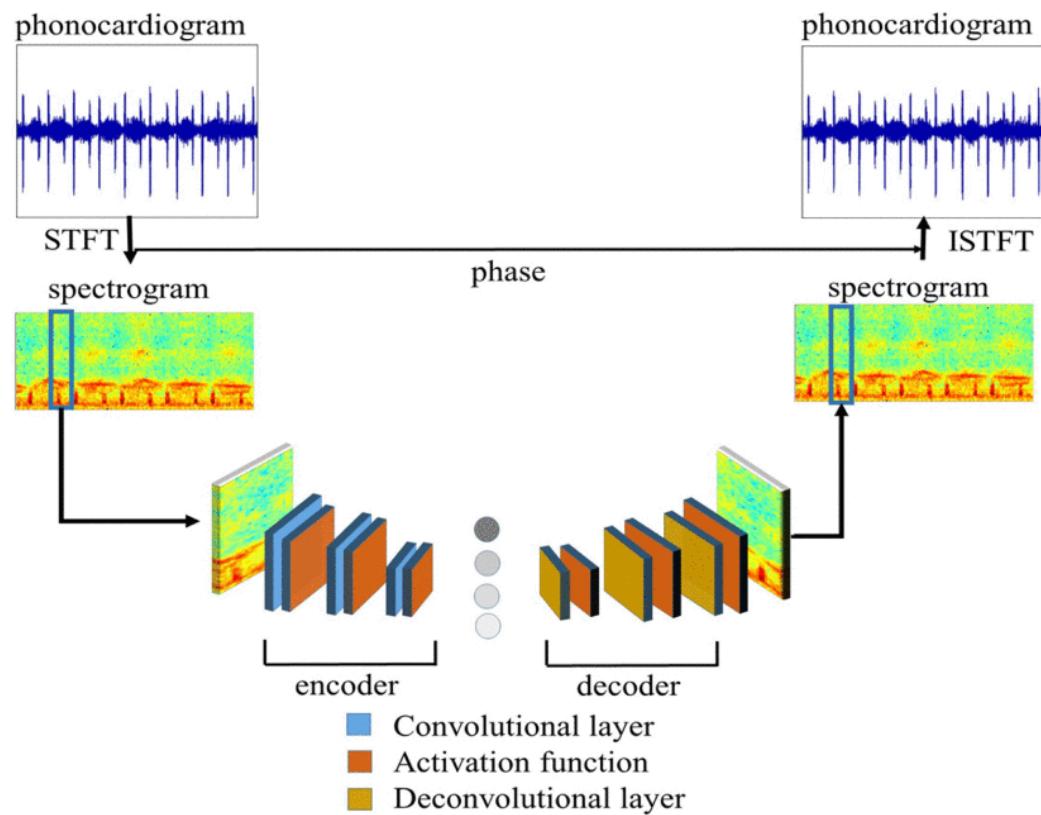


Figura 5.4: Analisi temporale e spaziale delle immagini di spettrogramma per essere usate come input SNN.

Capitolo 6

Risultati

La validazione di un sistema di raccomandazione rappresenta un passo fondamentale per verificare l'efficacia del modello creato. Tuttavia, la raccolta del feedback degli utenti rappresenta spesso un'impresa complessa e costosa. In questo studio, si è deciso di non coinvolgere direttamente gli utenti per ricevere un feedback sulle raccomandazioni fornite, ma abbiamo adottato delle strategie alternative per validare il sistema.

In primo luogo, si confrontano le raccomandazioni fornite dal sistema con le *playlist online* presenti su Spotify, che rappresentano una fonte di informazioni preziosa per comprendere le preferenze degli utenti e valutare l'efficacia delle raccomandazioni. In particolare, si è selezionato una serie di playlist che presentavano caratteristiche simili al dataset utilizzato, come ad esempio la presenza di brani di un certo genere musicale o di artisti specifici. Successivamente si è verificato se le raccomandazioni fornite dal sistema fossero presenti all'interno di queste playlist, valutando la corrispondenza tra i brani consigliati e quelli effettivamente presenti nella playlist.

Inoltre, si fa affidamento su *dataset di benchmark* per verificare la qualità delle raccomandazioni fornite dal sistema. Ad esempio, si è utilizzato il dataset **FMA** (Free Music Archive) e **Last.fm**[19], che contengono informazioni su migliaia di brani musicali e relative etichette di genere, per determinare quali coppie di canzoni sono state classificate come simili. Suc-

cessivamente, le informazioni raccolte sono state confrontate con le raccomandazioni fornite dal sistema, verificando la presenza di brani simili tra le raccomandazioni e le coppie di canzoni considerate simili dal dataset.

Infine, si è deciso di incorporare dei valori numerici percentuali che ci permettono di avere una visione quantitativa sull'efficacia del sistema di raccomandazione. Questi valori numerici permettono di valutare l'efficacia del sistema in maniera oggettiva e di individuare eventuali aree di miglioramento:

1. **Diversità**: la diversità è una misura della varietà delle raccomandazioni fornite dal sistema di raccomandazione. La diversità può essere calcolata utilizzando diverse metriche, come la diversità dell'insieme (la frazione di elementi unici nel set di raccomandazioni), la diversità dell'insieme delle coppie (la frazione di coppie di elementi unici nel set di raccomandazioni) o la diversità basata sulla distanza (la media della distanza tra le raccomandazioni). In generale, maggiore è la diversità delle raccomandazioni fornite dal sistema, maggiore è la probabilità che l'utente trovi una raccomandazione interessante.
2. **Novità**: la novità è una misura della capacità del sistema di raccomandare elementi nuovi o inaspettati agli utenti. La novità può essere calcolata utilizzando diverse metriche, come la novità dell'insieme (la frazione di elementi unici nel set di raccomandazioni rispetto a quelli già noti all'utente) o la novità basata sulla distanza (la media della distanza tra le raccomandazioni e gli elementi noti all'utente). In generale, maggiore è la novità delle raccomandazioni fornite dal sistema, maggiore è la probabilità che l'utente trovi una raccomandazione interessante e sorprendente.
3. **Serendipità**: la serendipità è una misura della capacità del sistema di raccomandare elementi che sono sorprendenti e inaspettati per l'utente. La serendipità può essere calcolata utilizzando metriche come la frazione di raccomandazioni inattese o la frazione di raccomandazioni che sono molto diverse dalle preferenze dell'utente. In generale, maggiore

è la serendipità delle raccomandazioni fornite dal sistema, maggiore è la probabilità che l'utente trovi una raccomandazione sorprendente e interessante.

4. ***Inattesa***: questa metrica valuta la frazione di raccomandazioni che sono inattese o sorprendenti per l'utente. In altre parole, questa metrica misura la capacità del sistema di raccomandare elementi che l'utente non si aspetta. Un sistema di raccomandazione che fornisce raccomandazioni inattese potrebbe migliorare l'esperienza dell'utente e aumentare la probabilità che l'utente interagisca con le raccomandazioni fornite.
5. ***Copertura***: questa metrica valuta la percentuale di oggetti disponibili che sono raccomandati dal sistema. In altre parole, misura la capacità del sistema di raccomandare un'ampia varietà di oggetti. Un sistema di raccomandazione con una copertura elevata può fornire all'utente una varietà di opzioni e aumentare la probabilità che l'utente trovi qualcosa che gli piace.

/	Diversità	Novità	Serendipità	Inattesa	Copertura
<i>BIRCH + Graph</i>	68.50 %	79.90 %	37.60 %	21.20 %	3.125 %
<i>SNN</i>	97.20 %	84.60 %	42.10 %	22.70 %	100 %

Figura 6.1: Esempio di analisi delle raccomandazioni fornite dal sistema di raccomandazione: per ciascuna metodologia considerata, sono state valutate le percentuali di Diversità, novità, serendipità e inattesa basate su un campione di 100 raccomandazioni. In totale, sono state esaminate 1000 canzoni, considerando che ciascuna raccomandazione include 10 brani.

6.1 Benchmark

Come già menzionato, per effettuare i confronti, vengono utilizzati i dataset di Last.fm e FMA. Questi dati sono stati generati da una pool di esperti del settore con competenze specifiche in ambito musicale, il che li rende affidabili e degno di fiducia.

Attraverso l'analisi di alcune playlist create dagli utenti di Spotify, sono stati ricavati dati numerici utilizzabili per valutare l'efficacia del sistema di raccomandazione creato. In particolare, tramite API sono state prese **100 playlist**, contenenti un totale di *2589 canzoni*, le quali sono state inserite nel database e utilizzate per la raccomandazione.

Durante la fase di confronto, è emerso che il sistema di clustering + grafo ha avuto una maggiore coerenza con le playlist create dagli utenti. È importante notare che le playlist rispettavano in larga parte lo stesso prompt di genere, per cui una selezione di raccomandazione più selettiva si è dimostrata più efficace.

In media, considerando che ogni playlist analizzata possiede 25 canzoni, il sistema di **clustering + grafo** riesce a ricreare la playlist originale tramite le raccomandazioni per il **68%** delle canzoni, ovvero in media 17 canzoni. D'altra parte, la **rete neurale siamese** ha una percentuale di riuscita del **44%**, ovvero in media riesce a ricreare solo 11 canzoni della playlist originale.

I dataset di benchmark utilizzati fornivano una serie di gruppi di canzoni simili a un determinato input, compresi gruppi di canzoni con possibili casi di plagio. Questi gruppi sono stati utilizzati per riempire un dataset di prova, al fine di valutare l'efficacia delle raccomandazioni fornite dal sistema.

Nel corso dei confronti, la rete neurale siamese (**SNN**) ha restituito una percentuale di similarità del **79%**, mentre il **clustering** ha ottenuto una percentuale del **42%**.

6.2 Confronto

6.2.1 Clustering + Grafo

Con questa metodologia, è chiaro che l'utilizzo del grafo delle adiacenze porta a un miglioramento delle prestazioni del sistema di raccomandazione. Questa tecnica consente di rappresentare i dati in modo compatto e di **bassa dimensionalità**, il che è particolarmente utile per la gestione di dataset molto grandi.

La rappresentazione compatta dei dati consente di ridurre i tempi di elaborazione e di rendere più efficiente il processo di ricerca delle raccomandazioni. Inoltre, la bassa dimensionalità dei dati consente di ridurre il rischio di overfitting e di migliorare la capacità del modello di generalizzare su nuovi dati.

6.2.2 SNN

L'implementazione di una Siamese Neural Network ha dimostrato diversi vantaggi rispetto ad altre tecniche di similarità. Ad esempio, ci ha permesso di utilizzare dati a più **alta dimensionalità** rispetto a quelle utilizzate con un algoritmo di clustering tradizionale seguite da un implementazione di un grafo. Inoltre, ha permesso di ottenere risultati più precisi e coerenti rispetto alle tecniche tradizionali, specialmente per i brani che non rientrano in una categoria musicale specifica. Inoltre, l'uso della **Softmax** come layer di output ha permesso di avere un'ampia scelta di brani simili, anche al di fuori di un unico cluster.

6.3 Grafica

6.3.1 Ricerca

Viene resa disponibile all'utente una funzione di ricerca all'interno della piattaforma, la quale consente di effettuare ricerche specifiche di brani e artisti. L'esito della ricerca viene visualizzato in un carosello, ovvero una successione continua di elementi, contenente i risultati disponibili all'interno del catalogo di Spotify.

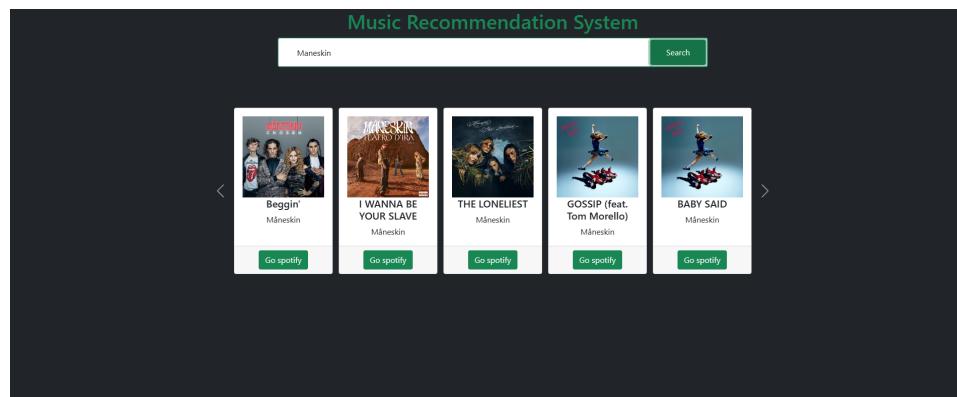


Figura 6.2: Questa immagine mostra un esempio di visualizzazione in cui viene effettuata una ricerca tramite “Maneskin”.

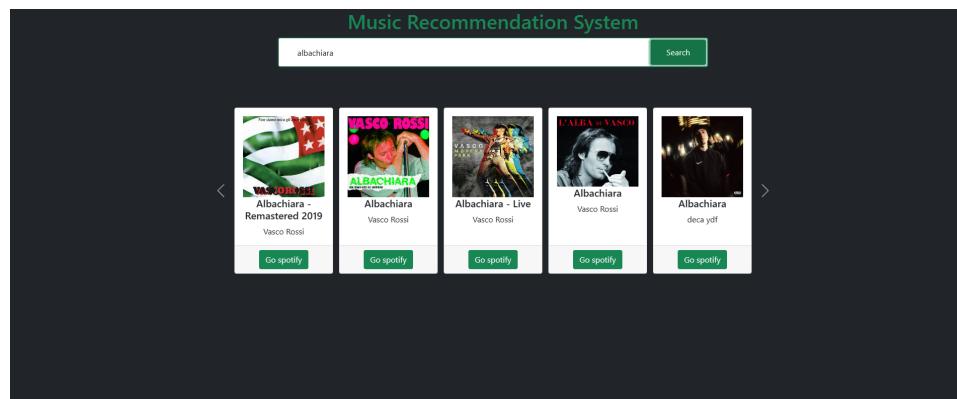


Figura 6.3: Questa immagine mostra un esempio di visualizzazione in cui viene effettuata una ricerca tramite “Albachiara”.

6.3.2 Raccomandazione

La piattaforma utilizza un sistema di raccomandazione basato sulla rete neurale artificiale (**SNN**). Durante l'esecuzione dell'algoritmo di raccomandazione, viene mostrato uno spinner di caricamento per indicare che il sistema sta elaborando le informazioni. Al termine di tale elaborazione, vengono visualizzate all'incirca 30 caselle rappresentanti le raccomandazioni fornite dal sistema. Queste caselle presentano la funzione di “mouse hover”, grazie alla quale è possibile visualizzare il titolo del brano, e con un semplice click è possibile ascoltare direttamente il brano su Spotify oppure un breve pezzo di brano in formato mp3.

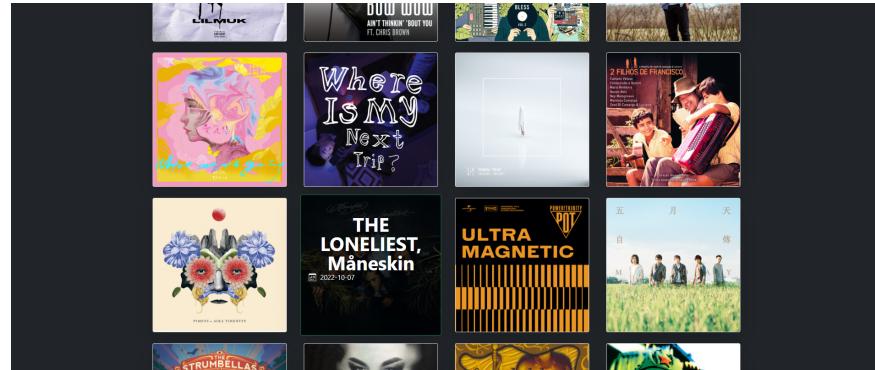
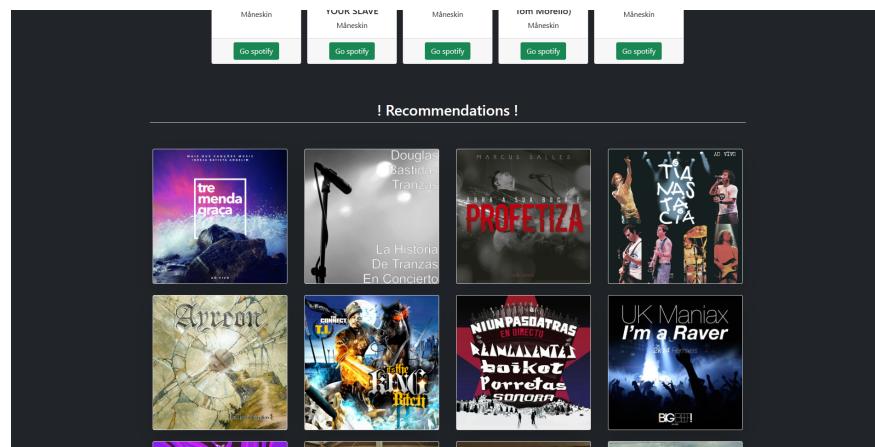


Figura 6.4: Gli esempi fanno riferimenti alla raccomandazione delle canzoni “Gossip” mostrato in Figura 6.2

Conclusioni

Le metodologie analizzate offrono spunti interessanti per migliorare l'efficienza dei sistemi di raccomandazione. L'utilizzo del grafo delle adiacenze in combinazione con il clustering, infatti, può portare a una rappresentazione più compatta dei dati, con tempi di elaborazione più veloci e una ricerca più efficiente delle raccomandazioni.

La bassa dimensionalità dei dati impiegati nel grafo delle adiacenze rappresenta un ulteriore vantaggio, in quanto riduce il rischio di overfitting e migliora la capacità del modello di generalizzare su nuovi dati.

L'implementazione di una Siamese Neural Network, grazie alla sua capacità di utilizzare dati a più alta dimensionalità, fornisce risultati più precisi e coerenti rispetto alle tecniche tradizionali. Inoltre, l'uso della Softmax come layer di output permette di ampliare la scelta dei brani simili, anche al di fuori di un singolo cluster.

In generale, l'applicazione di tecniche avanzate come il clustering, il grafo delle adiacenze e la Siamese Neural Network possono portare a miglioramenti significativi nelle prestazioni dei sistemi di raccomandazione, con una maggiore precisione nella selezione dei contenuti rilevanti per l'utente e una gestione più efficiente di grandi dataset.

Questo sistema di raccomandazione basato sull'analisi audio potrebbe essere utilizzato in diversi contesti, come ad esempio nei servizi di streaming musicale, dove gli utenti possono cercare raccomandazioni di brani o artisti simili ai loro gusti musicali. Inoltre, questo sistema potrebbe essere utilizzato per la ricerca di brani con somiglianze o plagi, come suggerito nell'ipotesi, con l'obiettivo di individuare eventuali violazioni dei diritti d'autore.

Per quanto riguarda gli sviluppi futuri, è possibile che questi sistemi di raccomandazione audio diventino sempre più sofisticati e personalizzati, adattandosi alle preferenze musicali specifiche dell'utente e integrando anche dati provenienti da altri contesti, come ad esempio i social media. Inoltre, potrebbero essere utilizzati anche per fornire suggerimenti in altri ambiti, come ad esempio nei servizi di podcasting o di audiolibri.

In merito alla questione del plagio, è importante notare che i sistemi di raccomandazione audio basati sull'analisi delle caratteristiche sonore di un brano potrebbero essere utilizzati come uno strumento di supporto per individuare eventuali somiglianze tra brani, ma non possono sostituire l'analisi approfondita e professionale di un esperto in materia di diritti d'autore.

In generale, l'evoluzione delle tecnologie di analisi audio e dei sistemi di raccomandazione potrebbe portare a nuovi sviluppi e applicazioni interessanti, ma è importante anche prestare attenzione alle possibili implicazioni etiche e di privacy.

La popolarità dei brani può essere un fattore importante da considerare nella raccomandazione di contenuti musicali. Alcuni utenti potrebbero preferire brani più popolari, mentre altri potrebbero essere alla ricerca di nuove scoperte musicali meno conosciute. Pertanto, è possibile utilizzare algoritmi di raccomandazione che tengano conto della popolarità dei brani, insieme ad altri fattori come la similarità e le preferenze personali dell'utente, per offrire un'esperienza di ascolto personalizzata e soddisfacente.

Appendice A

Appendice

Bibliografia

- [1] About youtube. [online]. <https://about.youtube.com/>. Accesso il: 09/01/2023.
- [2] About spotify. [online]. <https://www.truenumbers.it/statistiche-spotify>. Accesso il: 09/01/2023.
- [3] Sistema di raccomandazione di spotify. [online]. <https://www.arya-mohan.com/general-8>. Accesso il: 13/02/2023.
- [4] Sistema di raccomandazione di spotify. [online]. <https://newsroom.spotify.com>. Accesso il: 09/01/2023.
- [5] Spotify. (n.d.). spotify. [online]. <https://www.spotify.com/>. Accesso il: 09/01/2023.
- [6] Ek, d. & lorentzon, m. (2006). spotify [online music streaming service].
- [7] S. P. Chen e R. Jain. *Similarity-Based Pattern Recognition: An Algorithmic Approach to Cognitive Information Processing*. 2002.
- [8] Raghavan e Schütze Manning. *Introduction to Information Retrieval*. 2008.
- [9] Manhattan distance Wikipedia. https://it.wikipedia.org/wiki/Geometria_del_taxi. Accesso al sito il 24 febbraio 2023.

- [10] Chebyshev distance Wikipedia. https://it.wikipedia.org/wiki/Distanza_di_%C4%8Ceby%C5%A1%C3%ABv. Accesso al sito il 24 febbraio 2023.
- [11] Stuart P. Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.
- [12] R. Bromley, J. Zemel. *Siamese neural networks for one-shot image recognition*, pages 455–460. International Conference on Artificial Intelligence and Statistic, 1993.
- [13] Hadsell R. LeCun Y. Chopra, S. *Learning a similarity metric discriminatively, with application to face verification*. IEEE Conference on Computer, 2005.
- [14] Siamese Neural Network for images. <https://medium.com/swlh/one-shot-learning-with-siamese-network-1c7404c35fda>. Accesso al sito il 24 febbraio 2023.
- [15] Audio song similarity with SNN. <https://towardsdatascience.com/calculating-audio-song-similarity-using-siamese-neural-networks-62730e8f3e>. Accesso al sito il 24 febbraio 2023.
- [16] Spotify’s rate limit [online]. <https://developer.spotify.com/documentation/web-api/guides/rate-limits/>. Accesso il: 10/02/2023.
- [17] Michaël Defferrard, Kirell Benzi, Pierre Vandergheynst, and Xavier Bresson. FMA: A dataset for music analysis. In *18th International Society for Music Information Retrieval Conference (ISMIR)*, 2017.
- [18] Michaël Defferrard, Sharada P. Mohanty, Sean F. Carroll, and Marcel Salathé. Learning to recognize musical genre from audio. In *The 2018 Web Conference Companion*. ACM Press, 2018.
- [19] Last.fm. <https://www.last.fm/>. Accesso al sito il 14 febbraio 2023.

- [20] J. Chen, K. Li, Z. Tang, K. Bilal, S. Yu, C. Weng, and K. Li. A parallel random forest algorithm for big data in a spark cloud computing environment. *IEEE Transactions on Parallel and Distributed Systems*, 28(4):919–933, 2017.
- [21] G. Winskel and M. Nielsen. *Handbook of Logic in Computer Science (Vol. 4)*, chapter Models for Concurrency, pages 1–148. Oxford University Press, 1995.