Sistema di raccomandazione musicale

Tommaso Strada 829351
Sara Nocco 892030

13/09/2022

Sommario

L'obiettivo di questo progetto è quello di costruire un sistema di raccomandazione musicale per tutti coloro che vorrebbero imparare a suonare uno o più strumenti e per 'band' in cerca di nuovi brani da riprodurre.
Il sistema qui proposto permette di rintracciare le canzoni simili basandosi da un lato, su caratteristiche del brano nel suo complesso, dall'altro sulle peculiarità di ogni singolo strumento.
Vengono proposte in particolar modo due metriche: livello di difficoltà e rilevanza. L'architettura che garantisce il funzionamento del sistema di raccomandazione è di tipo 'grafo' ed è implementata su Neo4j.

Indice

1 Introduzione

Il progetto si divide in 4 parti: raccolta dati, analisi musicale, gestione dati e implementazione.
Nella prima fase vengono raccolti dati riguardanti informazioni sulla base musicale scelta come riferimento (Songsterr e FamousBirthdays), informazioni sulle caratteristiche delle canzoni nel complesso ('Spotify') e raccolta delle relative tracce audio.
Nella seconda fase si analizza ogni traccia audio con l'obiettivo di costruire due metriche per gli strumenti di batteria, basso e pianoforte: livello di difficoltà e rilevanza all'interno della canzone.
Le informazioni ottenute sono state processate al fine di rendere il dataset consistente e accurato. Nell'ultima fase si è costruito un modello a grafo che permettesse di sfruttare le relazioni tra dati per suggerire le canzoni più simili tra di loro.

1.1 Punti Principali

- Attributi brani: vengono ricercati e raccolti dati che arricchiscono le informazioni dell'utente sulla composizione del brano. Ad esempio gli artisti che hanno prodotto il brano o la tecnica utilizzata.
- Natura dei brani: vengono analizzati alcuni aspetti intrensechi dei brani come l'energia o la frequenza.
- Neo4j: la struttura del database viene scelta in modo da ottimizzare le relazioni tra dati.

2 Obiettivo

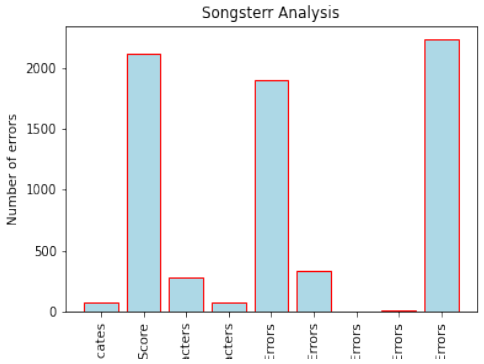
L'obiettivo del progetto è quello di fornire un database che permetta all'utente di trovare, brani simili per caratteristiche al brano di partenza. La ricerca dei match può essere fatta sia per il brano nel suo complesso che per i quattro strumenti principali che, se presenti, la compongono.
I quattro strumenti proposti sono: chitarra, batteria, basso e pianoforte.
Le informazioni riguardanti ciascun brano sono state selezionate con l'obbiettivo di aiutare l'utente ad acquisire un quadro completo della difficoltà e della rilevanza dello strumento all'interno del brano.
I brani proposti come affini a quello di partenza includono anche informazioni che possono aiutare l'utente nella prossima selezione, come i membri della band con il relativo strumento.

3 Raccolta dati

La raccolta dati avviene da tre fonti diverse: Songsterr, Spotify e FamousBirthdays. Di seguito vengono approfondite le metodologie utilizzate e i dati ottenuti.

3.1 Songsterr: Beautifulsoup

La base musicale è stata costruita partendo Songsterr, un noto sito web che offre canzoni e spartiti per imparare a suonare strumenti musicali.
La scelta è ricaduta su questo sito per la vasta offerta di brani e per la presenza di alcune features di rilievo come: difficoltà, artista compositore e tecniche utilizzate.
L'estrazione di dati è avvenuta mediante Beautifulsoup, una libreria python molto utilizzata nell'ambito di web-scraping.
Per utilizzare questa libreria è necessario disporre di pagine html da analizzare.
La versione più recente del sito web di Songsterr permette però una raccolta limitata di informazioni e più onerosa in termini computazionali a causa di elementi interattivi che rendono la struttura della pagina dinamica.
Si è quindi partiti dalla versione meno aggiornata da cui, una volta estratta la lista di tuning disponibili, sono stati generati 42 url attivi che riportassero a una lista di canzoni ordinate per popolarità e data di aggiunta.
Il motivo dietro questo passaggio è che ciascuna pagina, generata dalla selezione di una o più opzioni, contiene almeno un brano unico.
Infine dalla struttura html di ciascuna pagina sono stati estratti i link, riferiti a ciascun brano, che rimandassero alla versione web più aggiornata di songsterr.Da queste pagine sono state estratte le informazioni utili al progetto.
Come riportato sopra, stati generati 42 link contenenti mediamente 242 canzoni. Da questi sono stati estratti complessivamente 9960 link riferiti a brani di cui 2674 duplicati.
Dei rimanenti 7312 link, il 99,64% conteneva informazioni utili.
Partendo quindi da un dataset con 7286 righe e 6 colonne, si è passati ad un'esplorazione più approfondita dei dati a disposizione. Dalle analisi effettuate è emerso che vi erano 78 duplicati e 2110 righe senza livello di difficoltà. In entrambi i casi si è proceduto alla rimozione dei dati.
Successivamente è stata indagata la composizione degli attributi 'Band' e 'Song', da cui è emerso che il 7% è stato scritto con un alfabeto diverso da quello latino oppure con delle combinazioni di lettere prive di significato.
La presenza di questi caratteri è dovuta al fatto che Songsterr permette ai propri iscritti di caricare dei brani personalizzati o delle cover. Ai fini del progetto sono stati eliminati questi casi.
Si è proceduti infine alla formattazione dei dati per garantire consistenza interna.
Gli attributi ottenuti da questa fase sono: Band, Song, Player, Technique, Difficulty e Score.
I primi tre fanno riferimento al nome della band, della canzone e dell'artista che suona la chitarra. I rimanenti attributi fanno riferimento alla tecnica utilizzata, il livello di difficoltà stimato da Songsterr e il relativo punteggio numerico (su una scala da 1 a 8).



3.2 Spotify: API-Spotipy

Per ottenere features riguardanti la composizione musicale della canzone nel suo complesso come ad esempio l'energia (misura di attività ed intensità percepita) o bpm (l'unità di misura di frequenza) e altre informazioni riguardanti il gradimento del pubblico, si è fatto riferimento ai dati messi a disposizione da Spotify.

Per ottenere tali dati, si è reso necessario utilizzare Spotipy, un'apposita libreria python che permette, dopo avere ottenuto le API da Spotify, di accedere ai suoi database.
Sono stati forniti come input ad una funzione, gli attributi 'Band' e 'Song' di ciascun brano.
Sono stati restituiti i seguenti attributi:

Band : nome della band.
Song : nome della canzone.
ID TRACK : codice univoco del brano .
POPULARITY : popolairtà della band in una scala da 0 a 100.
RELEASE DATE : data di uscita del brano.
EXTERNAL URLS : URL che rimanda alla pagina Spotify contenente il brano.
SEARCH URL : URL contenente un campione del brano di 30 secondi.
AVAILABLE MARKETS : paesi in cui Spotify rilascia informazioni.
danceability : descrive quanto il brano è adatto ad essere ballato (in una scala da 0 a 1) .
loudness : rumorosità totale del brano (in dB).
speechiness : indica il livello di parole presenti (scala da 0 a 1).
acousticness : confidenza (tra 0 e 1) che il brano è acustico.
instrumentalness : indica se la traccia contiene parole.
liveness : individua la presenza di un pubblico.
valence : descrive la positività del brano.
tempo : frequenza stimata della canzone (bpm).
duration ms : durata in millisecondi.

Sono stati processati 4743 dati e restituite informazione solo per il 48.15 %.
Un ulteriore 10 % però non contiene l'attributo 'SEARCH URL' che risulta fondamentale per l'estrazione del file audio.
Il motivo di un risultato così basso è dovuto alla presenza di brani molto particolari, come cover e colonne sonore, la cui formattazione è difficilmente estendibile a tutto il dataset.
Parte del risultato è dovuto anche ad un'inefficienza di Spotify.

Per garantire un sistema quanto più omogeneo possibile è stato scelto di tenere solamente i brani con un valido.
Il dataset ora disponibile conta 1808 righe e 26 colonne.

3.3 FamousBirthdays: Selenium Beautifulsoup

Con l'idea di estendere il contenuto dell'attributo 'Player', contenente le informazioni sull'artista compositore del brano suonato tramite chitarra, agli altri strumenti presi in considerazione, è stata effettuata un'ulteriore operazione di web-scraping.
Le informazioni riguardanti i membri delle band sono state ottenute dal sito `http://www.famousbirthdays.com`
In questo caso è stata prima simulata la navigazione sul sito web mediante un driver, al fine di ricercare il nome della band nell'apposito spazio di ricerca e ottenere gli url alle pagine dei singoli artisti.
Successivamente sono state estratte le informazioni relative agli artisti mediante webscraping.
Sono state utilizzate rispettivamente le librerie Selenium e BeautifoulSoup.
Sono state aggiunte al dataset informazioni per 127 delle 806 band band presenti.
Laddove erano già presenti informazioni sull'artista compositore si è preferito considerare solo gli elementi ottenuti dallo scraping di Songsterr, ritenendolo una fonte più accurata.

4 Analisi Musicale: Librerie e metriche

4.1 Librerie: Librosa e Spleeter

L'obiettivo di questa fase è quello di ottenere informazioni sulla composizione musicale per ogni strumento, costruendo infine due nuove metriche: livello di difficoltà e livello di rilevanza.
Per fare ciò sono stati estratti dei sample delle canzoni da 30 secondi ciascuno.

Questi sono stati ricavati dall'attributo 'SEARCH URL' messo a disposizione da Spotify.
I 1807 brani sono stati analizzati mediante due librerie per l'analisi musicale: Spleeter e Librosa.
La prima permette di suddividere i file audio in 5 componenti: vocals, bass, piano, drums, others.
La seconda permette l'analisi musicale vera e propria.

4.2 Costruzione metriche: difficoltà e rilevanza

In particolar modo, mediante Librosa sono state ricavate per ciascun strumento e ciascun brano due metriche: Energy e BPM.
La prima rappresenta l'intensità del segnale.
La seconda invece è un'unità della frequenza.
Di queste metriche è stata calcolata la media, con relativa deviazione standard.
I valori sono stati infine normalizzati.
Sono stati assegnati dei livelli categorici per la variabile Energy e BPM nel seguente modo:

Valori > 0.66 = High
Valori <= 0.66 e > 0.33 = Medium
Valori < 0.33 = Low

Dalla combinazione di questi due livelli è stato calcolato il livello di difficoltà, secondo il seguente schema:

Energy Level	BPM Level	Difficulty Level
Low	Low	Beginner
Medium	Low	Beginner
High	Low	Beginner
Low	Medium	Intermediate
Medium	Medium	Intermediate
High	Medium	Intermediate
Low	High	Advanced
Medium	High	Advanced
High	High	Advanced

La rilevanza, invece è stata ottenuta, per ogni strumento, dal rapporto tra la media Energy dello strumento e 'energy' per l'intera canzone. Il valore così ottenuto è stato infine trasposto su una scala da 1 a 10.

5 Controllo qualità: Completezza e Consistenza

Di seguito vengono brevemente riassunte le operazioni di pulizia del dataset, eseguite e riportate nei punti precedenti, al fine di garantire consistenza interna tra gli attributi e completezza.

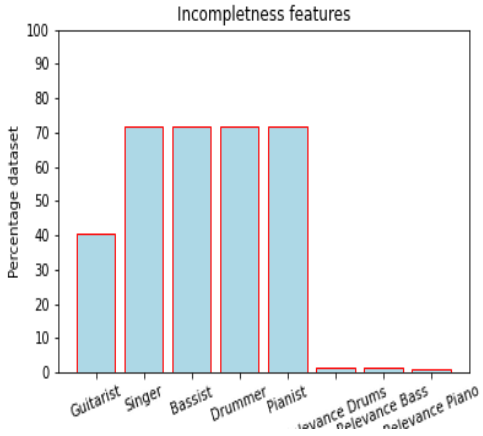
Consistenza : per garantire tale misura sono stati più volte controllati e rimossi i duplicati;
è stato uniformato l'attributo 'RELEASED DATE' tenendo esclusivamente l'anno di pubblicazione;
per l'attributo 'PLAYER' successivamente rinominato 'Guitarist' sono stati confrontati i dati ottenuti da Songsterr con quelli di FamousBirthdays e nel caso di conflitto sono stati considerati solo quelli della prima fonte;
sono stati rimossi i dati che presentavano caratteri di un alfabeto non latino (come quello cirillico) e caretteri privi di senso;
sono stati eliminate informazione aggiuntive presenti negli attributi 'Band' e 'Song' per migliorare la ricerca tramite spotipy.

Completezza : per aumentare la completezza degli attributi sono stati rimossi i dati che non avevano il corrispettivo file audio;
sono stati rimossi i dati senza valori dell' attributo 'Difficulty';

Dal dataset di partenza che contava 7286 elementi e 6 attributi, si è arrivati ad un dataset con 1763 elementi e 30 attributi.

5.1 Completezza Attributi

Oltre alle analisi di completezza riportate nel report è stata effettuata un'ulteriore analisi circa la completezza interna di tutti gli attributi.
Delle 53 colonne presenti, 8 presentano valori mancanti.
In particolare emerge una forte presenza di valori nulli tra gli attributi che descrivono i componenti delle band, come ad esempio 'Bassist' o 'Drummer', dove si raggiunge il 70 % di valori mancanti.
Di seguito vengono riportate le percentuali di dati mancanti:



5.2 Consistenza tra Energy ed energy

Infine è stata condotta un' analisi circa la consistenza interna tra il valore 'Energy' riferito ai singoli strumenti e 'energy' complessivo dell'intero brano.
Nonostante non sia noto il metodo di costruzione della metrica di Spotify 'energy', è stato assunto che il valore di energia prodotto da uno strumento non possa essere superiore all'energia totale del brano.
Effettuando un'analisi più approfondita è emerso che il 3 % degli elementi nel dataset presenta questo tipo di inconsistenza.
Di seguito vengono riportati gli elementi dello strumento pianoforte che presentano questa inconsistenza.

	Band	Song	energy	Energy_Mean_Piano_N
19	Ludwig Van Beethoven	Moonlight Sonata	0.00527	0.073230
31	Johann Sebastian Bach	Bouree	0.38500	1.000000
116	C418	Sweden	0.01000	1.000000
133	Red Hot Chili Peppers	Dark Necessities	0.74200	1.000000
328	Dream Theater	Lie	0.35100	0.391424
359	Therapy	Long Distance	0.20600	0.809530
382	Allen	Savior	0.02630	0.321492
416	Sens	Mon Ame	0.33900	0.437479
520	Chopin	Funeral March	0.00326	0.842743
717	Deftones	Anniversary Of An Uninteresting Event	0.38000	0.413520
1332	Merten	walk	0.71600	0.780680
1651	Rom	Coffee	0.16600	0.184241

6 Data Preparation and Storage

6.1 Neo4j e i motivi alla base della scelta del modello a grafo

Neo4j è un sistema di DBMS nativo per l'immagazzinamento di dati modellati con uno schema a grafo. Un modello a grafo è di tipo NoSQL, e gode perciò dei loro vantaggi e delle loro caratteristiche.
Tra queste, Neo4j soddisfa il cosiddetto principio BASE ed è inoltre caratterizzato dalle proprietà di consistency e availability (vale a dire che i dati in esso memorizzati sono coerenti tra i vari nodi ed il sistema è sempre disponibile).
In quanto modello NoSQL, Neo4j è un DBMS "schemaless", vale a dire che il suo schema è flessibile e che può essere costruito ed aggiornato basandosi sui dati da immagazzinare.
Ciò costituisce un enorme vantaggio e uno dei motivi principali alla base della scelta di questo tipo di modello a supporto dei dati a disposizione.
Un DBMS schemaless rappresenta infatti la soluzione migliore nella prospettiva di un arricchimento dei dati di tipo file audio, al fine di avere una base dati per il sistema di raccomandazione più ampia e offrire quindi una scelta più numerosa; inoltre, la flessibilità dello schema è imprescindibile per facilitare un

ampliamento dei dati per il miglioramento della accuratezza dei suggerimenti.
Infine, Neo4j, come tutti gli altri DBMS che supportano un modello NoSQL, si basa sull'assunzione di "mondo aperto".
L'ipotesi di "mondo aperto" si oppone a quella di "mondo chiuso", secondo la quale tutto ciò che non è presente nel database non esiste e la realtà rappresentata dai dati di interesse si esaurisce in quelli a disposizione.
Di conseguenza, l'assunzione di "mondo aperto" si adatta alla consapevolezza che i dati acquisiti non rappresentano l'interezza del fenomeno di interesse, dal momento che potenzialmente si potrebbero introdurre altre proprietà, oltre a quelle selezionate, per descrivere una traccia musicale, un artista/band e infine per caratterizzare la difficoltà di esecuzione tramite uno strumento.
La modellazione dei dati tramite grafo è motivata dallo scopo finale per il quale sono stati acquisiti i dati, ovvero la creazione del sistema di raccomandazione.
Il modello a grafo è tipicamente il modello alla base di sistemi di raccomandazione in quanto questi ultimi sfruttano molteplici proprietà, caratterizzate da relazioni articolate, nonché molteplici livelli di relazione.
Ed è proprio a partire dalla relazione tra canzone e strumento-livello di difficoltà associato che si sviluppa il sistema di raccomandazione ivi costruito.

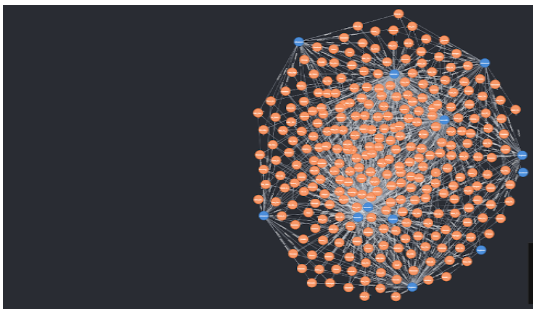
6.2 Processamento dati e upload su Neo4j

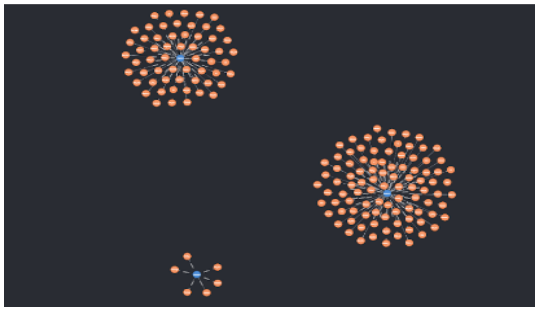
La modellazione della struttura a grafo è stata implementata su Python tramite la generazione di un file di testo.
Tramite una serie di cicli che prendono in input gli opportuni valori, ogni riga del file di testo utilizza la clausola CREATE, in linguaggio Cypher, per la creazione di ogni nuovo nodo e di ogni nuova relazione. In particolare, ogni istanza creata ha un codice alfanumerico identificativo univoco, generato iterativamente.
Le librerie Python utilizzate in questa fase sono: re, pandas, numpy.
Sono stati definiti due diversi tipi di nodi:

Track : rappresentante una canzone e contenente le caratteristiche ad essa associate che possono essere di interesse per il suo suggerimento. Le caratteristiche ('properties') selezionate sono: 'Band', 'Song', 'Singer', 'POPULARITY', 'RELEASE DATE', 'danceability', 'energy', 'loudness', 'speechiness', 'acousticness', 'instrumentalness', 'liveness', 'valence', 'tempo', 'duration ms'.

Instrument Difficulty" : rappresentante la coppia di strumento presente nella canzone e relativo livello di difficoltà. Le proprietà inserite sono: 'instrument', 'difficulty'.

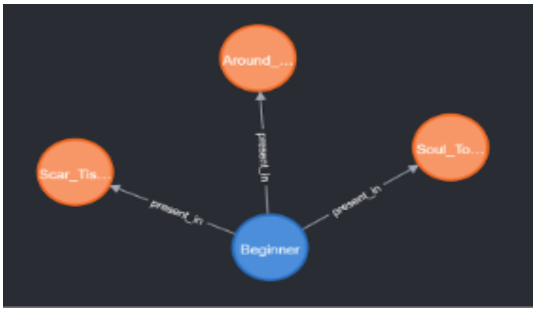
I due tipi diversi di nodi sono collegati tra loro da un arco, che ne definisce la relazione. Ogni arco è dotato di due proprietà, a caratterizzare ulteriormente il legame tra brano e strumento-livello di difficoltà.
In particolare, per gli archi colleganti un brano alla chitarra (qualsiasi sia il livello di difficoltà associato), sono stati rappresentati la tecnica con cui il brano è eseguito ('Technique') e il chitarrista da cui il brano è stato originariamente eseguito.
Per gli archi colleganti un brano ai restanti strumenti (ancora una volta, indipendentemente dal suo livello di difficoltà), sono stati rappresentati invece il livello di rilevanza dello strumento all'interno della canzone e il musicista che ha originariamente eseguito il brano.
La figura sottostante mostra il grafo risultante dalla struttura appena descritta.
Sono riportati i primi 300 nodi. I nodi blu sono i nodi di tipo "Instrument Difficulty".





La struttura così definita permette di eseguire queries di vario tipo, andando a sfruttare e filtrare le richieste a seconda delle properties di interesse.
Di seguito è riportata una query scritta in linguaggio Cypher, con relativo output, per avere tutte le canzoni della band Red Hot Chili Peppers (informazioni contenute nel nodo "Track") caratterizzate da una difficoltà per l'esecuzione della batteria di livello "principiante".

MATCH(song:TrackBand:'Red Hot Chili Peppers')<-[r:present in]-(strum:Instrument Difficultyinstrument:'drums', difficulty:'Beginner') RETURN strum, r, song



Un esempio di query che va a sfruttare le proprietà riportate negli archi è la seguente: sono restituite tutte le canzoni eseguite con la chitarra per mezzo della tecnica "overdriven", indipendentemente dal livello di difficoltà.

MATCH(song:Track)<-[r:present intechnique:'OverdrivenGuitar']-(strum:Instrument Difficultyinstrument:'guitar') RETURN strum, r, song

7 Conclusioni

L'obbiettivo del progetto è stato raggiunto.
Il sistema di raccomandazione costruito permette efficacemente di ritrovare canzoni con caratteristiche simili, per i 4 principali strumenti musicali, fornendo inoltre ulteriori informazioni.
La struttura implementata su Neo4j permette di effettuare queries con alta velocità di risposta e potrebbe essere utilizzato in applicazioni web.
Partendo da questo risultato si potrebbe aumentare la libreria musicale a disposizione colmando l'assenza di file audio di Spotify, con altre fonti, quali ad esempio Youtube.
Inoltre si potrebbe effettuare uno scraping più approfondito per estrarre informazioni circa la composizione delle band, dove la percentuale di incompletezza risulta essere elevata.

8 Sitografia

Songsterr Official url : https://www.songsterr.com/
Songsterr Unofficial url : https://www.songsterr.com/a/wa/all?r=tuning&lyrics=any&tuning=any&diff=any&inst=gtr&sort=p&vocals=any
FamousBirthdays Official url : http://www.famousbirthdays.com
Spotify documentation : https://developer.spotify.com/
Spotipy documentation : https://spotipy.readthedocs.io/en/master/

Librosa documentation : https://librosa.org/doc/latest/index.html
Spleeter documentation : https://github.com/deezer/spleeter
Neo4j documentation : https://neo4j.com/
Beautifulsoup documentation : https://www.crummy.com/software/BeautifulSoup/bs4/doc/
Selenium documentation : https://www.selenium.dev/documentation/