**DOCUMENTAZIONE CASO DI STUDIO**

**INGEGNERIA DELLA CONOSCENZA 2023-2024**

**Progetto realizzato da:**

**-Alessandro Olivieri, matricola: 703269,** [**a.olivieri14@studenti.uniba.it**](mailto:a.olivieri14@studenti.uniba.it)

**-Antonio Silvestre, matricola: 697697, a.silvestre2@studenti.uniba.it**

**Link allo zip del progetto:**

**Indice**

1. Introduzione
2. Dataset e Features
3. Modelli di classificazione
   * K-Nearest Neighbors
   * GaussianNB
   * Random Forest
   * Extra Trees
4. Ottimizzazione degli Iperparametri
   * Curva di validazione
   * Exhaustive grid search
5. Accuratezze dei Classificatori
   * Metriche di Valutazione
   * Risultati
6. Considerazioni Finali
7. Clustering
   * Algoritmo KMeans
   * Valutazione con Purity

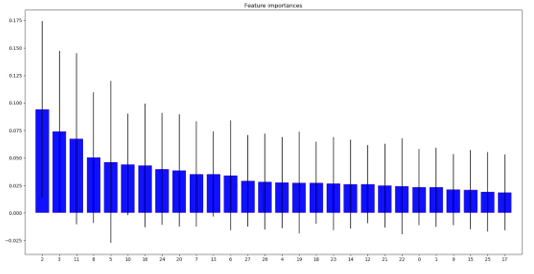
**1. Introduzione**  
L’obiettivo di questo progetto è di far riconoscere a partire da uno specifico tipo di traccia audio, che è il WAV, il genere, attraverso il confronto e la valutazione di alcuni dei principali modelli di classificazione basati su apprendimento supervisionato che sono: KNN (K-Nearest Neighbors Classifier), Naive Bayes Classifier, Random Forest Classifier & Extra Trees Classifier.

Si è notato come le performance (precisione, accuratezza e richiamo) dell’Extra Trees Classifier siano risultate superiori, in generale, rispetto agli altri modelli.

I generi che dovrà riconoscere sono: blues, classical, rock, hip hop, reggae, disco, country, metal, pop, jazz e ciò sarà possibile attraverso feature estraibili direttamente da tracce audio.  
  
**2. Dataset e Features**:  
Il dataset che è stato utilizzato è: (https://www.kaggle.com/datasets/insiyeah/musicfeatures) chiamato “music featuers” che contiene 1000 esempi di file audio di cui 800 per il training e 200 per il test. Le feature presenti nel dataset, per ogni brano sono: (ottenute da specifiche tecniche di estrapolazione dai file)

* **Tempo (0):** Misura dell’andamento della musica
* **Beats (1):** Unità ritmica della musica
* **Chroma\_stft (2):** Short Time Fourier Transform
* **RMSE (3):** Root Mean Square Error
* **Spectral\_centroid (4):** Indica dov’è posizionato il “centro di massa” dello spettro
* **Spectral\_bandwith (5):** Misura dell’ampiezza d’onda dello spettro il cui valore non può essere minore della metà del suo massimo
* **Roll-off (6):** L'attenuazione via via crescente delle frequenze gravi o acute man mano che ci si allontana dalla gamma centrale.
* **Zero\_crossing\_rate (7):** Velocità con cui il segnale cambia da positivo a negativo, o viceversa
* **MFCCs (8-27):** Mel-frequency cepstral coefficients (MFCCs) sono i coefficienti che insieme identificano l’MFC (sono stati presi in considerazione 20 coefficienti)
* **Label (Target Feature):** Nome del genere associato al file audio

Queste features sono fondamentali per una traccia audio.   
Durante la fase di pre-processing del dataset, si è dedotto che, ognuna delle caratteristiche risulta penalizzante, perciò non è stato possibile ridurre il numero delle features.



**3. Modelli di classificazione**

Per ottenere una predizione sui nuovi esempi, sono stati utilizzati più modelli di classificazione basati su apprendimento supervisionato, ottenuti dalla libreria sklearn poiché l’obiettivo era di valutare l’accuratezza di ogni singolo modello in fase di test

* *K-Nearest Neighbors*:

è un algoritmo utilizzato nel riconoscimento di pattern per la classificazione di oggetti basandosi sulle caratteristiche dei k oggetti più vicini a quello considerato. Un oggetto è classificato in base alla maggioranza dei voti dei suoi *k* vicini. *k* è un intero positivo tipicamente non molto grande. La scelta di k dipende dalle caratteristiche dei dati. Generalmente all'aumentare di k si riduce il rumore che compromette la classificazione. Al fine dell’apprendimento lo spazio multidimensionale viene partizionato in regioni in base alle posizioni e alle caratteristiche degli oggetti di apprendimento, rappresentati come vettori. Un oggetto è assegnato alla classe C se questa è la più frequente fra i k esempi più vicini all'oggetto sotto esame, la vicinanza si misura in base alla distanza fra punti. I vicini sono presi da un insieme di oggetti per cui è nota la classificazione corretta.

* *GaussianNB*:

I classificatori basati sul modello Naïve Bayes, utilizzano il teorema di Bayes:

Dove *P(y|x1,…,xn)* è la probabilità a posteriori, *P(y)* è la probabilità a priori, *P(x1,…,xn|y)* è la verosimiglianza e *P(x1,…,xn)* è la funzione di partizione. Nell’ utilizzo del classificatore GaussianNB, si presume che la probabilità delle feature sia gaussiana:

I parametri *σy* e *μy* sono stimati usando la massima probabilità.

* *Random Forest:*

È un modello d'insieme ottenuto dall'aggregazione tramite bagging di alberi di decisione. Esso è un meta-stimatore che si adatta ad una serie di alberi decisionali addestrati su vari sotto-campioni del dataset e utilizza la media di ogni singolo output di ogni albero per migliorare l’accuratezza predittiva e il controllo del sovradattamento. Il Random Forest deve essere dotato di due matrici: una matrice X sparsa che contiene i campioni di addestramento e una matrice Y di dimensioni che contiene i valori target.

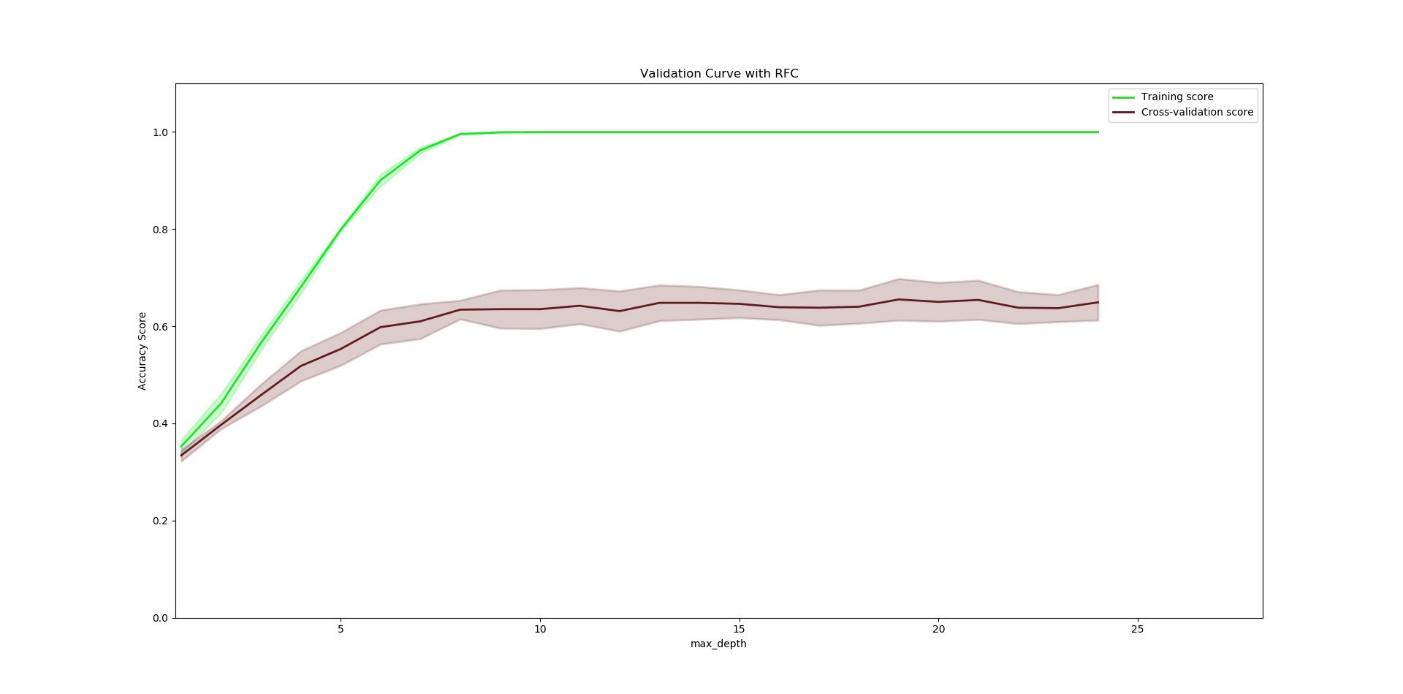
* *Extra Trees:*

Tale modello è simile al precedente, la differenza risiede nella scelta degli alberi, la quale avviene in maniera puramente casuale.

**3. Ottimizzazione degli iperparametri**

Si è eseguito un processo di ottimizzazione degli iperparametri con lo scopo di rendere i vari modelli di classificazione più accurati.  
Inizialmente in ogni modello di classificazione sono inclusi parametri con specifici valori, detti iperparametri.

Se questi ultimi non vengono espressi, ai parametri si assegneranno valori standard di default, che non aiutano il modello ad accrescere la propria accuratezza.  
Per stabilire gli iperparametri, si è fatto uso di diverse tecniche di ottimizzazione ovvero:  
  
***Curva di validazione*:**

metodo vantaggioso per evidenziare i valori teoricamente ottimizzati di ciascun modello.  
Essa può essere disegnata su un grafico, per mostrare il comportamento che il modello assume cambiando i valori di un singolo iperparametro.  
  


Tale grafico, mostra come avviene la validazione per il parametro “max depth” presente all’interno del modello di classificazione “ExtraTrees Classifier”.  
Sull’asse delle ascisse vi è il parametro che si vuole settare in base ai valori che può avere, sulle ordinate vi è il valore di accuratezza.  
Le curve di training score e cross-validation score raffigurano il concetto fondamentale di questa procedura, infatti, in base a queste si può verificare per quale valore dell’iperparametro, diventa massima l’accuratezza.  
  
***Exhaustive grid search*:**    
 Questo metodo genera i possibili iperparametri attraverso una griglia di valori definita dal parametro “param\_grid”, formato da una gamma di valori per ogni parametro definito dall’utente.  
Vengono valutate le varie combinazioni di assegnazioni degli iperparametri in modo da scegliere quella migliore e mostrarla al termine del processo.  
  
Grafici ottenuti: (

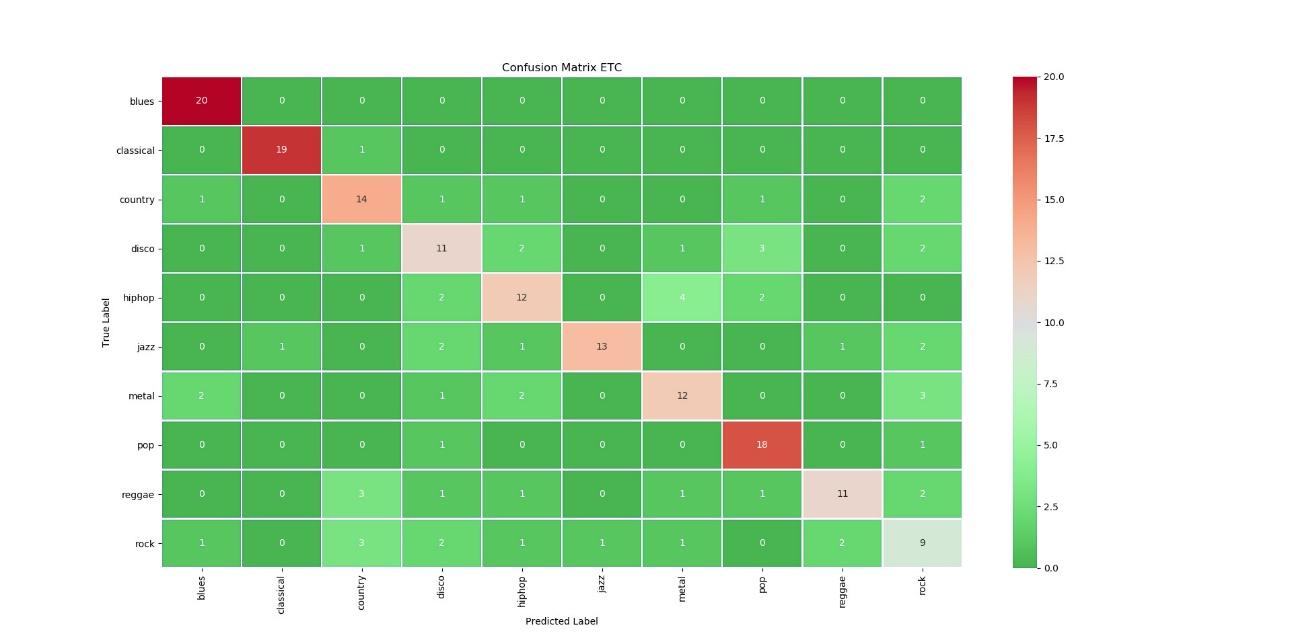
**4. Accuratezze classificatori**  
Gli iperparametri di ogni classificatore sono stati utilizzati per predire la classe corretta di una precisa traccia.  
In basso verranno riportati tutti i valori delle metriche usate con lo scopo di testare l’efficienza di ogni classificatore.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Classificatori | Accuracy | Precision | Recall | F1-measure |
| *K-NN* | 0.365 | 0.371 | 0.365 | 0.356 |
| *GaussianNB* | 0.435 | 0.404 | 0.435 | 0.389 |
| *RandomForest* | 0.530 | 0.535 | 0.530 | 0.505 |
| *ExtraTrees* | 0.670 | 0.665 | 0.670 |  |

**5. Considerazioni**

Dopo aver confrontato i vari classificatori, il modello che si è scelto di utilizzare è l’ExtraTreesClassifier poiché riesce a predire maggiormente quale sarà il genere di una traccia.

* **Confusion matrix Extra Trees Classifier:**



* **Metriche utilizzate Extra Trees Classifier:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Genere | Precision | Recall | F1-score | Support |
| Blues | 0.92 | 0.60 | 0.73 | 20 |
| Classical | 0.91 | 1.00 | 0.95 | 20 |
| Country | 0.46 | 0.30 | 0.36 | 20 |
| Disco | 0.56 | 0.75 | 0.64 | 20 |
| Hiphop | 0.71 | 0.75 | 0.73 | 20 |
| Jazz | 0.61 | 0.70 | 0.65 | 20 |
| Metal | 0.73 | 0.80 | 0.76 | 20 |
| Pop | 0.70 | 0.80 | 0.74 | 20 |
| Reggae | 0.65 | 0.55 | 0.59 | 20 |
| Rock | 0.58 | 0.55 | 0.56 | 20 |

**6. Clustering**

È stata effettuata una procedura di clustering basata sull’algoritmo *KMeans*. Tale algoritmo ha come scopo quello di minimizzare la varianza totale intra-gruppo, e nel nostro caso è stato utile per raggruppare, in cluster diversi, brani appartenenti a generi musicali diversi.

Prima di procedere, abbiamo portato sulla stessa scala le varie input features.

Una volta eseguito il clustering, è stato possibile valutare i risultati grazie alla metrica denominata *purity*. Per calcolare questa metrica, ogni cluster è assegnato alla classe che è più frequente nel cluster, e successivamente l’accuratezza di questa assegnazione è misurata contando il numero di tracce assegnate in modo corretto, e dividendo per il numero di esempi.

In particolare, la purity è rappresentata da un punteggio nell’intervallo [0,1], che indica in quale misura ognuno dei cluster contenga una ed una sola classe. Di conseguenza, ad un valore maggiore di tale metrica corrisponde una maggiore omogeneità dei cluster, e dunque un partizionamento in classi più accurato.

